Global Under-Resourced MEedia Translation (GoURMET)

H2020 Research and Innovation Action
Number: 825299
D5.5 – GoURMET Final progress report on integration

Nature	Report	**Work Package**	WP5
Due Date	30/06/2022	**Submission Date**	30/06/2022
Main authors	Sevi Sariisik Tokalac (BBC), Dominic Tinley (BBC), Peggy van der Kreeft (DW)		
Co-authors	Wilker Aziz (UVA), Anna Blaziak (BBC), Lei He (BBC), Jindřich Helcl (UEDIN), Vivek Iyer (UEDIN), Antonio Valerio Miceli Barone (UEDIN), Juan Antonio Pérez-Ortiz (UA), Guillem Ramirez Santos (UEDIN), Víctor Sánchez-Cartagena (UA), Felipe Sanchez-Martínez (UA), Martin Valchev (BBC)		
Reviewers	Barry Haddow (UEDIN)		
Keywords	integration, MT, API, translation, monitoring, content creation, demonstrator		
Version Control			
v0.1	**Status**	Draft	20/06/2022
v1.0	**Status**	Final	29/06/2022

Contents

For ease of cross-referencing, the section numbering in this document follows the same structure as deliverable D5.3 Initial Integration Report as far as practicable. Where work was completed in the first half of the project, there are some sections where there is nothing new to report, but this information can be found by looking for the equivalent numbered section in D5.3.

Also, where GoURMET languages are listed, they are generally included in order of development with the numbering consistent across this and D5.6 Final Evaluation Report (e.g. details about Macedoniam, language 9, can generally be found in subsection X.9 or X.X.9).

List of Figures

Abstract

This deliverable describes how the GoURMET translation models developed by the research partners have been integrated and trialled in multilingual newsroom settings. It focuses on the prototypes developed by the media partners BBC and Deutsche Welle (DW) to integrate the machine translation (MT) models developed into real life workflows, assesses the extent goals for each use case was realised, and relays feedback from newsroom journalists on the usefulness of the GoURMET models in particular and MT in general for broadcast media applications.

1 Introduction

1.1 WP5 overview

This document forms part of a series of deliverables that describes how the GoURMET translation models developed by the research partners have been integrated and trialled in a multilingual newsroom setting.

Work Package 5, coordinated by the British Broadcasting Corporation (BBC) News Labs Multilingual Journalism team comprised five tasks:

T5.1 requirements gathering – see D5.2 Use Cases and Requirements

T5.2 creation of shared interfaces – see D5.3 Initial Integration Report

T5.3 platform integration and deployment – some reported in D5.3 and more in this deliverable

T5.4 media monitoring user evaluation – see D5.4 Initial Evaluation + D5.6 Final Evaluation

T5.5 global content creation user evaluation – see D5.4 Initial Evaluation + D5.6 Final Evaluation

This document focuses on the prototypes developed by the media partners BBC and Deutsche Welle (DW) to showcase the translation models developed in the project and to obtain feedback from newsroom journalists on their usefulness. Full details of earlier work can be found in the deliverables listed above. The following sections provide a brief overview of earlier work for context:

1.2 Use cases overview

D5.2 Use Cases and Requirements describes three overall project use cases, namely:

1. Global Content Creation
2. Media Monitoring
3. International Business News Analysis

In this report, in order to reflect the logical order of the media workflows (i.e. focus on content discovery first, and then pursue further content creation) we will first discuss Media Monitoring, followed by content creation.

The third overall project use case aimed to focus on an under-resourced domain, rather than an under-resourced language, and was adapted in the later stages of the project. Rather than tackling the business domain, GoURMET consortium partners agreed to address health which offered more immediate, relevant and wider-reaching opportunities for both the BBC and DW. This decision and respective outcomes are described further in sections 6.3.1 and 8.3.

1.3 Potential benefits overview

D5.2 Use Cases and Requirements goes on to describe more detailed 'media partner use cases', which would be more accurately described as 'potential benefits'. Five outcomes were specified in the report for the BBC:

A. Improving internal visibility

B. Increased workflow efficiency for reversioning output

C. Editorial oversight

D. Media insight

E. Research and experimentation with semi-automated content production

Two were specified for DW:

F. Translation and Adaptation for Content Creation

G. Translation for Cross-Lingual Media Monitoring

The prototypes were shaped with a view to addressing as many of these more detailed use cases, or potential benefits, as possible. For further details see sections 6, **??** and 10.

1.4 Platform overview

The platform integration activities in WP5 covered a number of processes. A conceptual summary of the overall WP5 integration activities can be seen in Figure 1.

The primary purpose of these activities was to take the output of the research partners, indicated in red on the right-hand side of Figure 1, and make those translation models available to the generic integrations, indicated in grey on the left-hand side of the diagram, using a translation service. It is worth noting that the API itself has been released on an open source licence.

1.5 Prototypes overview

During the second half of the project, the focus moved from developing underlying infrastructure to developing prototypes that utilised this infrastructure in order to gauge the usability and quality of machine translation (MT) in a production environment.

1.5.1 BBC

The BBC developed three prototypes to address the three overall use cases:

1. Live Pages Translation addressed use case 2 (monitoring) – see 6.1
2. Frank addressed use cases 1 and 2 (monitoring and content creation) – see 6.2
3. The Multlingual Graphical Storytelling Tool addressed updated use case 3 (health) – see 6.3

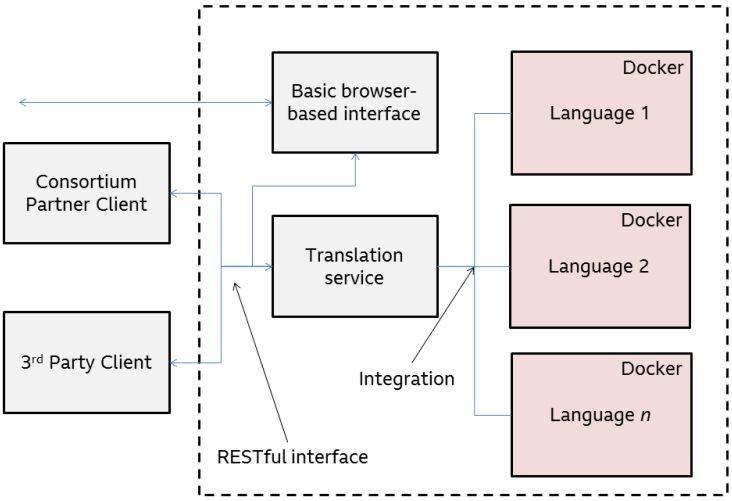

Figure 1: Conceptual summary of WP5 integration activities

1.5.2 DW

DW built three prototypes to address the three overall use cases:

1. plain X focuses on use case 1 (content creation) – see 7.1
2. SELMA addresses uses case 1 and 2 (media monitoring and content creation) – see 7.2
3. The DW Benchmarking Tool provides continuous quality assessment for all use cases – see 7.3

2 Translation Model Delivery and Integration

Translation Model Delivery and Integration was initiated during the first part of the project and a full description of the early work is available in D5.3 Initial Integration Report. During the second part of the project several new or revised translation models came on stream, and there was extensive work to resolve issues with some of the earlier models.

2.1 Building a Compliant Docker Image

See D5.3 section 2.1

2.2 Integrate.py

See D5.3 section 2.2

2.3 Integration API

See D5.3 section 2.3

2.4 Performance tests

2.4.0 Context

Prototypes developed by the BBC and DW under the scope of the GoURMET project served a dual purpose. They provided:

1. A platform to evaluate the models in circumstances as close as possible to real life
2. A means to expose multilingual journalists from the BBC and DW to machine assisted translation workflows, enhance solutions based on feedback, and advocate for future strategic investments in the field.

As the leading user partner for integration, the BBC conducted a series of 'needs and opportunities' interviews and surveys with BBC World Service staff at month 24 of the project as part of an internal handover process at the point that several BBC staff left and others joined the GoURMET project.

At this stage we detected a significant degree of mistrust of and resistance to machine learning solutions among journalists. This was a concern as we wanted to ensure that journalists' impressions and experience of machine-assisted workflows under the GoURMET project were as positive as possible.

To ensure that GoURMET was well received, in line with the original project aspirations, the BBC concluded that the quality of the models presented to journalists should be either on a par with or better than other available translation services that journalists would be familiar with (e.g. Google Translate).

Since the work to conduct rigorous evaluations of each translation model would not be completed until later in the project, the BBC conducted preliminary performance tests of each model to decide which languages to integrate with each prototype. The criteria for selection were:

1. Speed
2. Quality
3. Engagement

Some translation models performed slower than others, and some were not sufficiently accurate for practical use at this early stage of the project. As in any case it would not be practical to seek feedback from every language team at the BBC, the team prioritised the translation models that performed most effectively.

2.4.1 Speed

We conducted a comparison of each model's speed using the setup in Table 1

Task definition: A task definition is required to run docker containers in Amazon ECS (see docs.aws.amazon.com/AmazonECS/latest/developerguide).

Table 1: Speed test setup

Compute Engine	Task Definition	Task Memory (GB)	Task CPU (vCPU)	Task GPU (vGPU)
AWS Fargate	1	4	2	N/A

Task Memory: The amount of memory used by the task. It can be expressed as an integer using MiB,(e.g. '1024') or as a string using GB (e.g. '1 GB').

Task CPU: The number of CPU units used by the task. It can be expressed as an integer using CPU units (e.g. '1024') or as a string using vCPU (e.g. '1 vCPU').

Test setup: This test uses the GoURMET Translate API.
(see https://translate-api.gourmet.newslabs.co/v1/translate).

We ran speed tests for each translation model five times and generated the average response time. Table 2 shows the average response time for each model ranked from slowest to fastest.

The tests were run against the deployed GoURMET Translation API rather than the models themselves on a local machine, but the other parts of the API only adds minimal latency so this is a true representation of the speed of the models.

While the primary focus of GoURMET is research, the intended destination for the translation models is fast-paced, real-life newsrooms. This dictates that integration decisions consider not just quality, which remains the core prerequisite, but also the speed of the models and cost of running them.

As described in D5.1 section 4.2.4 (Translation Speed) and D5.2 section 4.1.2 (Non-Functional Requirements) translating 'a maximum time of 500ms per sentence of 80 words maximum is set as a minimum requirement' to ensure efficient usability. However, testing in the second half of the project (M23) indicated that the GoURMET models were not readily matching this speed on the architecture being. This meant that while the models were feasible for use as prototypes, further work would be needed to make them practical for wider use. There were a few things we could do to help with this:

- Increase CPU and memory – we tried this for some models
- Design the architecture of the client app to account for the speed of the models (e.g. using WebSockets to load pages one part at a time – we tried this approach with the first prototype LPT)
- Run the translation as a background job (e.g. pre-translate the articles) – we tried this approach with the second prototype Frank
- Run more ECS tasks simultaneously – we also tried this approach with the second prototype Frank

In an ideal world, we would not need to complicate the architecture in a client app with background tasks in order to make translation work. However, by adopting this approach, translation speeds of 15 words/s or above can be made to work for the purposes of prototyping and testing.

Table 2: Speed test result

Model	Words per second
en→yo	1.68
en→tr	2.94
yo→en	3.03
tr→en	3.09
gu→en	4.34
en→gu	5.29
en→ps	5.93
my→en	7.09
bg→en	7.34
en→ig	8.35
en→ha	8.87
ta→en	9.79
ps→en	10.38
ky→en	11.04
am→en	11.06
en→ta	11.26
en→ky	12.37
ig→en	13.11
en→mk	13.36
ha→en	13.73
mk→en	15.59
en→sw	17.39
en→am	18.3
sr→en	19.17
en→my	19.34
en→ti	22.68
ti→en	23.29
en→bg	23.52
sw→en	24.16
en→sr	25.76
en→ur	37.53
ur→en	51.42

We considered if the issues with speed could be resolved by using a GPU (graphics processing unit) rather than CPU (central processing unit) compute engine. The cloud service that had been selected for us early in the project, Amazon Web Services (AWS) Fargate, does not support GPU. This was investigated as part of a deeper look at issues with the Swahili model which are explained in more detail in section 8.1.

However, while GPU offers enhanced mathematical computational capability, which is a benefit for machine-learning tasks, GPU computing in the cloud is more costly. It is also not straightforward to scale a GPU-based platform quickly as when launching additional instances it can take several minutes for the appropriate containers to be loaded. This is a particular issue when usage is low, as it is for our prototypes, because it is not cost effective to have multiple instances running just in case they're needed, and the usage trends are less smooth than they are for translation systems operating at a massive scale (e.g. Google Translate).

We have reflected that the speed requirements set out in D5.2 are ambitious targets when balanced against the need to keep costs to a minimum (i.e. not to be keeping computing power permanently available on standby in anticipation of it being used). As we go on to explain, the models as currently deployed are more compatible with some use cases than others, but with some retrospective snagging and performance optimisation, we have been able to make the prototypes work sufficiently well for user testing and evaluation of all the scenarios we wished to explore.

To develop what we have from prototypes to production tools will require more detailed consideration of the requirements in order to balance cost, quality and speed (e.g. it is possible that with greater uptake, the cost per use can be brought down and the case for keeping GPU instances running permanently may become stronger), The work conducted to date will be valuable in framing this analysis to move the tools to the next level.

2.4.2 Quality

Since evaluation results described in D5.6 Final Evaluation were not available at the point the integration process started, we initially took a very basic approach to assessing quality in their absence.

We conducted small-scale sample comparisons using the GoURMET models available at the time each prototype was developed. We selected samples from the BBC websites (e.g. a generic political stories which have high incidence, such as elections) and translated these into a target languages using GoURMET.

We then back-translated the output using GoURMET and Google, in order to assess the translation directly – observing whether or not the quality was acceptable for contextual use.

Two example of the results are shown below:

2.4.2.1 Serbian example

English taken from BBC website: Thailand and Montenegro are being added to the UK government's red list – meaning they are considered to be among the highest-risk destinations.

Serbian translation by GoURMET: Tajland i Crna Gora nalaze se na crvenoj listi britanske vlade – što znači da se one smatraju jednim od najrizičnijih destinacija.

Serbian translation back to English by Google: Thailand and Montenegro are on the red list of the British government – which means that they are considered one of the most risky destinations.

Serbian translation back to English by GoURMET: Thailand and Montenegro are on the British government's red list, which means they are considered to be one of the most risky destinations.

Conclusion: Acceptable translation to render meaning.

2.4.2.2 Hausa example

English taken from BBC website: Thailand and Montenegro are being added to the UK government's red list – meaning they are considered to be among the highest-risk destinations.

Hausa translation by GoURMET: Ana kara samun kasashen Thailand da Montenegro da ke cikin jerin kasashen da gwamnatin ta UEMOA ke amfani da su – ma'ana ana daukarsu a matsayin cikin kasashe da ke da karfin tattalin arziki.

Hausa translation back to English by Google: Thailand and Montenegro are increasingly on the list of countries used by the UEMOA government – meaning they are considered to be among the world's most powerful economies.

Hausa translation back to English by GoURMET: Thailand and Montenegro are increasingly on the list of countries used by the UEMOA government – meaning they are considered to be part of the US-led coalition.

Conclusion: In both cases the back translations were inaccurate. Testing with further examples indicated that the model was not reliable for deployment at this stage.

Although this method is not optimal, our findings from these exercises broadly correlated with the data from the evaluations.

2.4.3 Engagement

The models needed to perform well in terms of speed and accuracy to be reliable and usable in a prototype (e.g. following a developing news event from a source language, or speeding up long translations). However, these were not the only factors to consider to bring about successful, sustainable prototypes. A third is the level of engagement from the end-user representatives, particularly editorial managers.

The models might perform perfectly in isolation for sentence segments, but if they deteriorate when applied to the kind of content or context that journalists need them for (e.g. batches of content, news formats, tight turnaround times) or with the tooling available for deployment due to scalability reasons (e.g. long waits for pages to load), then user engagement declines.

Furthermore, in certain cases where the models did provide an *accurate* translation but it was not deemed acceptable under the *stylistic expectations* of the relevant team, users were unwilling to engage and commit to continued usage.

Some examples of these came up in interviews with Chinese, Arabic and Russian editorial teams. While these were not GoURMET languages per se, we were particularly interested in the views of these teams as they represented 'parent hubs' whose content was extensively monitored and

reused by other teams that shared geographical or cultural interests. In each case they were also well-resourced languages with extensive, good quality commercial models.

We asked the teams about their experience of machine translations of content from their language into English, and from English into their language. The consensus was that translations into English seemed useful and correct overall. However, the quality of translations from English into their target languages was considered unsatisfactory, particularly for longer, more complex sentences and contexts. A frequent response was that the language sounded 'mechanical', 'idiomatically odd', or 'unnatural', and that it did not provide a pleasurable reading experience.

One senior journalist commented that Arabic needed to sound poetic, adding that 'One word can make a big difference. In one instance I've seen that the machine translated the name of a person and changed it into a common noun.' [In this case the name in Arabic meant 'Moon' and it was translated literally into the word 'moon'].

One Chinese editor said 'I dislike machine translation – it is difficult to rewrite all the content, it gives ideas and context but it is difficult to correct it to make it publishable. I need to first understand what the machine is trying to say and then translate it to normal [sic] language… I don't want people to think that I'm lazy using the machine translation.'

A Russian digital editor commented that they do not use translations in their output, barring exceptional cases. In this case their hypothesis is that translated content is inferior for journalism, and that it is always preferable to find a journalist to research and write a story directly into the target language in order for it to sound authentic.

In order to ensure sustainable utilisation of the models in the long run, we needed to offer an improvement to daily workflows in the form of prototypes. Several meetings were held with team editors of the languages with 'acceptable and above par' performance to explain the wider GoURMET project goals and the specific proposals, and to obtain their feedback to ensure any prototype served their needs and made it worthwhile for them to invest time and effort.

A further consideration was the appetite (e.g. willingness and commitment) of editorial stakeholders to release team time to engage in prototype trials which would enable the compilation of post-edit data to serve as 'gold standard' evaluation of the models.

The timing of the gold standard evaluation was a particular challenge. Due to a combination of amended work patterns under Covid-19 measures, an extensive restructuring within BBC News that led to serious staff shortages, and the added pressures of the Russia-Ukraine War, the editorial teams prioritised their 'must-have outputs' and were very unwilling to release team members to engage with the trials. Further details of how we worked around this issue are provided in the D5.6 Final Evaluation Report.

2.5 Languages shortlisted and integrated

There were a total of 16 languages selected for development over the course of the GoURMET project as shown in Figure 2.

A key consideration for choosing these languages, as illustrated by the map, was cultural, social and geographical proximity, so that the models could pave the way for better editorial compliance, monitoring and content exchange opportunities between such 'hub' regions where one country might be served by broadcasts in several languages.

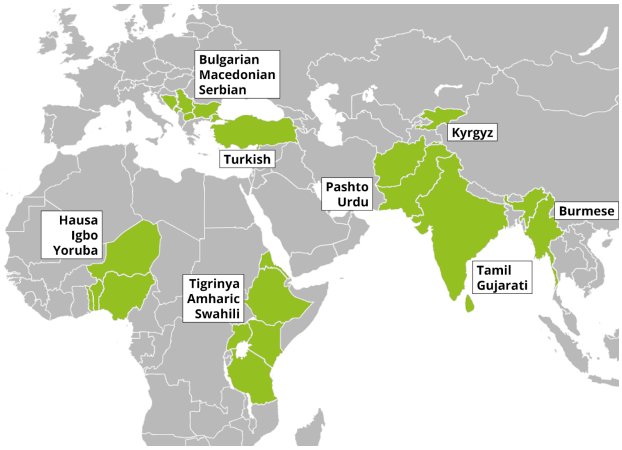

Figure 2: Map showing GoURMET languages as of 2022

Based on the three considerations already outlined in section 2.4, of speed (see 2.4.1), quality (see 2.4.2), and engagement (see 2.4.3), the following nine GoURMET languages (each language paired with English) were selected for integration in the BBC prototypes, listed here in the order of in which they were developed:

1. Swahili
5. Tamil
6. Serbian
10. Hausa
11. Igbo
12. Tigrinya
13. Pashto
16. Urdu
17. Turkish v2
18. Turkish v2+ health adaptation

In the case of Tigrinya, the quality was not on the same level as the other selected languages. However, Tigrinya was not available from commercial providers such as Amazon or Google at the time of the work conducted (but has since been released by Google). Therefore, it was useful to include it to enable other teams to view stories from the Tigrinya Service (e.g. reporting first hand on the circumstances of the Tigray Conflict that has been unfolding since 2020).

The translation models varied in their readiness for deployment to live media systems as they were developed by different academic partners who took different experimental approaches, used different methodologies and who were, depending on the nature of each language, subject to different linguistic constraints.

2.5.1 Languages shortlisted and integrated by BBC

Based on the considerations described above, Tables 3 and 4 show the languages that were integrated into each BBC prototype. Key points to note are that:

- All languages were integrated into the Demonstrator User Interface
- All languages that passed the BBC performance test were integrated into at least one additional BBC prototype

Table 3: BBC language implementation. UI, LPT and Frank use GoURMET translation models to translate the specified language to and from English. GST uses GoURMET translation models to translate the specified language to English

Language	UI	LPT	Frank	GST
1. Swahili	✓		✓	✓
2. Gujarati	✓	Insufficient quality		
3. Turkish	Used v2		Used v2	Used v2+
4. Bulgarian	✓	Not a BBC language		
5. Tamil	✓		✓	
6. Serbian	✓	✓	✓	✓
7. Amharic	✓	Insufficient quality		
8. Kyrgyz	✓	Insufficient quality		
9. Macedonian	✓	Not a BBC language		
10. Hausa	✓			✓
11. Igbo	✓		✓	✓
12. Tigrinya	✓	✓		
13. Pashto	✓		✓	
14. Burmese	✓	Insufficient quality		
15. Yoruba	✓	Insufficient quality		
16. Urdu	✓		✓	
17. Turkish v2	✓		✓	Used v2+
18. Turkish v2+				✓

2.5.2 Languages shortlisted and integrated by DW

All GoURMET models were integrated in at least one DW application. The integration of the GoURMET models by DW depended on the prototype and its intended use. The SELMA research prototype, for instance, covers all GoURMET languages. For productive use, the GoURMET

Table 4: BBC language implementation (simplified list). UI, LPT and Frank use GoURMET translation models to translate the specified language to and from English. GST uses GoURMET translation models to translate the specified language to English

UI	LPT	Frank	GST
All	6. Serbian 12. Tigrinya	1. Swahili 5. Tamil 6. Serbian 12. Tigrinya 13. Pashto 16. Urdu 17. Turkish v2	1. Swahili 6. Serbian 10. Hausa 11. Igbo 18. Turkish v2+

models of the current DW languages were targeted in the first place and within that group, those that score best, or those that score better than the other engines were integrated with priority.

Thus, not necessarily the comparative rating among GoURMET models, but the comparison between the GoURMET model and other third-party engines that are available in the prototpye is decisive for implementation.

2.5.3 Additional machine translation models

2.5.3.1 Providers included by the BBC

The prototypes developed by the BBC aimed to bring as many of the BBC News language services together as possible in order to optimise the benefits. Therefore, commercially available models were deployed alongside GoURMET models to ensure the widest possible coverage and utility.

We took the approach that to keep the usability of the tools as simple as possible each language should only be served by one model, selected on the basis of speed and quality, rather than offering multiple models per language which journalists must choose between. This approach requires upfront decisions about usage before the deployment stage,

In order to aid decisions on this point as well as language selection for GoURMET to target underserved languages, the portfolios of major providers were examined side by side with a view to select one provider to work with. There were four criteria considered for the decision:

1. Coverage
2. Cost
3. Quality
4. Ease of integration

When the first prototype was being developed in 2020, it used Google models as these were deemed the best available in terms of coverage and quality. By July 2021, when work started to build BBC's second prototype, Frank, portfolios of all major commercial providers were compared based on publicly available information for enterprise solutions. At this point in time, BBC World Service was broadcasting in 43 languages (see Tables 6 and 7).

Table 5: DW language implementation. All prototypes use GoURMET translation models to translate the specified language to and from English.

Language	**DW**	**plain X**	**SELMA OSS**	**Bench marking**
1. Swahili	✓	✓	✓	✓
2. Gujarati			✓	
3. Turkish	✓	✓	✓	✓
4. Bulgarian	✓	✓	✓	✓
5. Tamil	✓	✓	✓	✓
6. Serbian	✓	✓	✓	✓
7. Amharic	✓	✓	✓	✓
8. Kyrgyz			✓	
9. Macedonian	✓	✓	✓	✓
10. Hausa	✓	✓	✓	✓
11. Igbo			✓	
12. Tigrinya			✓	
13. Pashto	✓	✓	✓	✓
14. Burmese			✓	
15. Yoruba			✓	
16. Urdu	✓	✓	✓	✓
17. Turkish v2	✓	✓	✓	✓
18. Turkish v2+	✓			

Considering each criterion in turn:

1. **Coverage**
   There was not a significant difference in coverage between the three large commercial providers of translation models (Google, Amazon, Microsoft) – the difference is negligible, particularly when GoURMET languages are added to fill the gaps – see Tables 6 and 7.
2. **Cost**
   The costs associated with each commercial provider are in similar ranges – there are not significant savings to be achieved with one provider over another.
3. **Quality**
   Sample comparison exercises were conducted at various points in the process to compare GoURMET, Amazon, Microsoft and Google translations, as well as smaller scale tasks involving translation platform providers RWS/Trados and Smartcat (see paragraph below).
4. **Ease of integration**
   The final consideration was about technical architecture and ease of complementing the models with GoURMET and the API.

As an example of quality, one round of RWS sampling involved seven relatively high-resourced languages (Portuguese, Swahili, Spanish, Russian, Arabic, Persian, Turkish). For each language, three articles of approximately 300 words were selected. One article for each language was drawn from the political domain from a kind of story where the wording is frequently reproduced (e.g. news about upcoming national elections). These stories were combined with one event with global resonance (e.g. in this case the controversy around Novak Djokovic's Covid-19 test), and one about science or technology, so that the samples' content and difficulty levels were more or less similar across languages. These were then compared against Google output in both directions. In this particular exercise, the comparison texts totalled more than 17K words. The goal was to be able to identify the kind of output that is both as accurate as possible and sounds as close to the language's natural style and news mannerisms as possible when consumed in isolation.

For the second prototype, Frank, the team planned to include Amazon Translate languages alongside the GoURMET models. The idea was to complement the experience of the first prototype, LPT, where we had used Google Translate. It was also an opportunity to investigate whether using Amazon models within an architecture built around Amazon Web Services (AWS) might make integration simpler or provide other advantages.

However, during the first sprint, a decision was taken to revert back to the Google models. This was because we had a lot of rate limit issues with Amazon Translate for which we were unable to find a simple solution. We also arranged a trial of the tool with the Igbo team, which is a language supported by Google but not Amazon.

2.5.3.2 Providers included by DW

The prototypes developed by Deutsche Welle aimed to have as wide a coverage as possible for both the content creation as well as the media monitoring use cases. From the start, it was decided to include commercially available models, including Google and Microsoft Azure to ensure a broad range of languages. This is then complemented by other service providers that have an added

Table 6: BBC World Service languages matched with providers in July 2021 (breakdown)

BBC	GoURMET	Google	Amazon	Microsoft
Afaan Oromoo				
Amharic	Amharic	Amharic	Amharic	Amharic
Arabic		Arabic	Arabic	Arabic
Azeri		Azerbaijani	Azerbaijani	Azerbaijani
Bengali		Bengali	Bengali	Bangla
Burmese	Burmese	Burmese		Myanmar
Chinese		Chinese (Simplified)	Chinese (Simplified)	Chinese (Simplified)
UK China		Chinese (Traditional)	Chinese (Traditional)	Chinese (Traditional)
Dari			Dari	Dari
English		English	English	English
French		French	French	French
Gujarati	Gujarati	Gujarati	Gujarati	Gujarati
Hausa	Hausa	Hausa	Hausa	
Hindi		Hindi	Hindi	Hindi
Igbo	Igbo	Igbo		
Indonesian		Indonesian	Indonesian	Indonesian
Japanese		Japanese	Japanese	Japanese
Kirundi		Kinyarwanda		
Korean		Korean	Korean	Korean
Kyrgyz	Kyrgyz	Kyrgyz		
Marathi		Marathi		Marathi
Nepali		Nepali		Nepali
Pashto	Pashto	Pashto	Pashto	Pashto
Persian		Persian	Persian	Persian
Pidgin				
Portuguese		Portuguese	Portuguese	Portuguese (Brazil & Portugal)
Punjabi		Punjabi		Punjabi
Russian		Russian	Russian	Russian
Serbian	Serbian	Serbian	Serbian	Serbian (Cyrillic & Latin)
Sinhala		Sinhala (Sinhalese)	Sinhala	
Somali		Somali	Somali	
Spanish		Spanish	Spanish & Spanish (Mexico)	Spanish
Swahili	Swahili	Swahili	Swahili	Swahili
Tamil	Tamil	Tamil	Tamil	Tamil
Telugu		Telugu	Telugu	Telugu
Thai		Thai	Thai	Thai
Tigrinya	Tigrinya			Tigrinya
Turkish	Turkish	Turkish	Turkish	Turkish
Ukrainian		Ukrainian	Ukrainian	Ukrainian
Urdu	Urdu	Urdu	Urdu	Urdu
Uzbek		Uzbek	Uzbek	
Vietnamese		Vietnamese	Vietnamese	Vietnamese
Welsh		Welsh	Welsh	Welsh
Yoruba	Yoruba	Yoruba		

Table 7: BBC World Service languages matched with providers in July 2021 (totals)

	GoURMET	Google	Amazon	Microsoft
BBC languages covered	14	39	32	33
BBC languages missing	29	4	11	10
BBC languages missing if other provider complemented by GoURMET		3	7	6

value, for instance DeepL (high quality, but not that many languages), eTranslation (free for public broadcasters, Facebook (open source and locally installable) and, of course, GoURMET (locally installable and focused on low-resource languages). Other providers specialised in specific regions can be added if required.

In particular for the plain X platform, we aim at a wide range of providers and languages, but it has to have some added value, i.e. something not yet covered by the other services in the platform (e.g. regional coverage, reduced cost, high quality) We implement a recommendation system, so that the best and/or most appropriate system is suggested or set as default for a specific language (pair).

3 Translation service system architecture

The Translation service system architecture was developed during the first half of the project and a full description of the early work to develop it is available in D5.3 Initial Integration Report.

During the second half of the project, no significant changes were made to the architecture.

3.1 Detailed system architecture

See D5.3 section 3.1

3.2 Security, access management and request rate limiting

See D5.3 section 3.2

3.3 Scalability

See D5.3 section 3.3

Table 8: DW languages matched with providers - June 2022 (breakdown)

DW	GoURMET	Google	Facebook	Microsoft
Albanian		Albanian	Albanian	Albanian
Amharic	Amharic	Amharic	Amharic	Amharic
Arabic		Arabic	Arabic	Arabic
Bengali		Bangla	Bangla	Bangla
Bosnian		Bosnian	Bosnian	Bosnian
Bulgarian	Bulgarian	Bulgarian	Bulgarian	Bulgarian
Chinese Simplified		Chinese Simplified	Chinese	Chinese Simplified
Chinese Traditional		Chinese Traditional	Chinese	Chinese Traditional
Croatian		Croatian	Croatian	Croatian
Dari				Dari
English	English	English	English	English
French		French	French	French
German		German	German	German
Greek		Greek	Greek	Greek
Hausa	Hausa	Hausa	Hausa	
Hindi		Hindi	Hindi	Hindi
Hungarian		Hungarian	Hungarian	Hungarian
Indonesian		Indonesian	Indonesian	Indonesian
Macedonian	Macedonian	Macedonian	Macedonian	Macedonian
Pashto	Pashto	Pashto	Pashto	Pashto
Persian		Persian	Persian	Persian
Polish		Polish	Polish	Polish
Portuguese for Africa		Portuguese	Portuguese	European Portuguese
Portuguese for Brazil		Portuguese	Portuguese	Portuguese
Romanian		Romanian	Romanian	Romanian
Russian		Russian	Russian	Russian
Serbian	Serbian	Serbian	Serbian	Serbian (Cyrillic & Latin)
Spanish		Spanish	Spanish	Spanish
Swahili	Swahili	Swahili	Swahili	Swahili
Tamil	Tamil	Tamil	Tamil	Tamil
Turkish	Turkish	Turkish	Turkish	Turkish
Ukrainian		Ukrainian	Ukrainian	Ukrainian
Urdu	Urdu	Urdu	Urdu	Urdu

Table 9: DW languages matched with providers in June 2022 (totals)

33	GoURMET	Google	Facebook	Microsoft
DW languages covered - including English	11	32	32	32
DW languages missing	22	1	1	1
DW languages missing if other provider complemented by GoURMET		1	1	0

3.4 Deploying machine translation models

While no changes were made to the architecture described in the deliverable D5.3 Initial Integration Report, some minor changes were made to the AWS Fargate task configuration to enable the deployment of later languages models, in particular for those of larger sizes (e.g. Pashto (9 GB) and Turkish v2 (12 GB)).

One change was to increase the amount of ephemeral storage allocated beyond the default amount (20 GB) set for tasks hosted on AWS Fargate. Another change was to increase the CPU and memory for AWS Fargate tasks and to increase the number of AWS Fargate tasks (e.g. 4 tasks for Tamil and Tigrinya where the default is 1).

A further change was to adjust two parameters of the models, BEAM_SIZE and BATCH_SIZE, to speed up translation. Both these variables are optional with default values 5 and 32 respectively. Changing the BEAM_SIZE to 1 gave faster results with some impact quality, reducing the BLEU score on the BBC and DW tests by no more than 1 BLEU point (e.g. for English to Pashto the scores with the BBC test set moved from 18.56 to 17.53 and with the DW test set from 12.52 to 12.05).

As well as making configuration changes, the architecture of the prototypes themselves was also adapted to work more smoothly with the models as more was learned about their strengths and limitations and about the GoURMET Translation API. For example, to improve the user experience for the Live Pages Translation prototype, pages were broken into sections with lazy loading via a WebSocket connection.

For example, instead of sending all articles in a stream to be translated at once, the translations of individual articles were requested synchronously, based on the availability of the translation provider. In the case of GoURMET, this meant limiting the amount of articles processed at a given moment per language. As a result, journalists could begin reading articles from different language streams as soon as possible, while additional articles loaded in the background, increasing the perception of speed and usability.

For further detail about the development of the Live Pages Translation prototype see section 6.1.

4 Translation API

The Translation API was developed during the first part of the project and a full description of the early work to develop it is available in D5.3 Initial Integration Report.

During the second part of the project no changes were made to the API.

4.1 Standards for translation APIs

See D5.3 section 4.1

4.2 Security

See D5.3 section 4.3

4.3 API details

See D5.3 section 4.4

4.4 Versioning

See D5.3 section 4.4

5 Demonstrator User Interface

The Demonstrator User Interface (UI) was developed during the first half of the project and a full description of the early work to develop it is available in D5.3 Initial Integration Report. During the second half of the project there were some minor changes to note.

5.0 Basic features of UI

See D5.3 section 5

5.1 Changes made to UI

Only one design change was made to the UI in the second part of the project which was to add support for languages written in the right-to-left aligned Perso-Arabic script (e.g. Pashto and Urdu). There were also some minor bug fixes required as new models were added:

- Timeouts – we changed the way these were managed to ensure requests for larger translations were completed correctly
- Default language – we fixed an issue that resulted in the UI not correctly storing the target language specified by the user and therefore returning translations in the first language on the list
- Missing languages – we resolved an issue that led to existing languages being deleted as new languages were added

5.2 Usage of UI

Figure 3 shows the GoURMET Translation API usage specifically from the UI. The ‘consumed quota’ shows the number of requests to the API on any given day.

Despite the dissemination efforts and some peaks in usage, overall monthly public usage of public UI was limited.

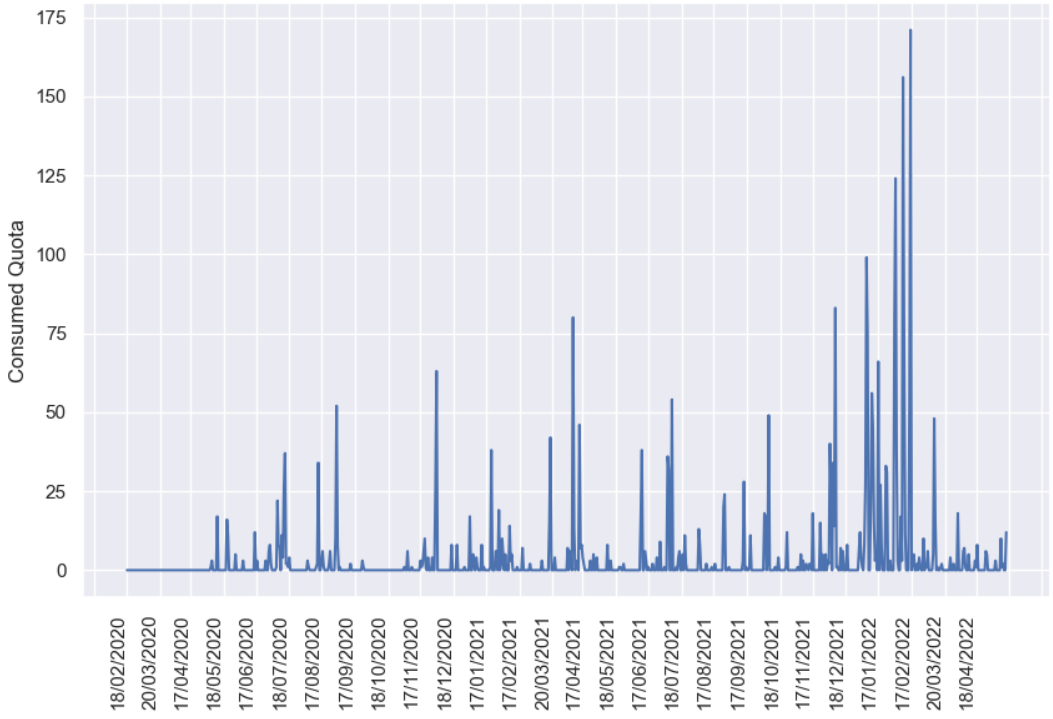

Figure 3: UI usage

5.3 Findings and Learnings

The chronological, incremental nature of the development process has provided the project team an extended learning experience about the requirements and scope of MT solutions to be deployed on or in tandem with live media production systems.

However, the way the tasks were distributed across the timeline meant that the architecture for the API and UI were developed without properly factoring in the specifications of the models that were yet to be developed. In hindsight, specifications such as speed, size, batch processing should have been raised, discussed and agreed with the consortium partners as part of the development process. The deployment and integration effort involved trying to adapt models with varying specs and structures to fit together.

Having a mid-point handover for both the editorial and the technical leads of the project at the BBC also had an adverse effect on the extent of insight and prior knowledge derived over the course of the first 18 months. The incoming team needed time to understand the tasks at hand (T5.3, T5.4, T5.5 in the Grant Agreement) to develop responses reactively. By the time the challenges of the API structure were identified, there had already been a great deal of investment, so to restart the process and experiment with new deployment approaches would have been inefficient.

One of the primary incentives for using in-house translation models instead of external providers is cost-efficiency. However, as things stand, the deployment infrastructure of models proposed by the BBC diminishes the potential for effectively reducing translation expenses.

Table 10: Number of characters translated using GoURMET models by month according to BBC API figures from 7 January to 20 June 2022

Month	Characters translated
2022 Jan	81452109
2022 Feb	91437138
2022 Mar	102275663
2022 Apr	107858251
2022 May	157647413
2022 Jun	126411989

The idea behind using AWS Fargate to deploy GoURMET models is good in theory but does not work so well in practice. In an ideal AWS Fargate configuration, tasks scale up horizontally to meet high translation demands and subsequently scale back down, potentially to zero. However, due to large docker image sizes and caching limitations, scaling up can often take more than a few minutes.

Effectively, one must keep at least one instance per language pair running continuously to offer on-demand translation capabilities. The cost of a single translation direction can range from 50 to 170 USD a month, depending on the hardware requirements for running the model. However, this estimate is a fixed monthly cost, regardless of translation demand.

Table 10 shows the current volume of translations made via the GoURMET API. As expected, the number of translated characters goes up as more and more languages are deployed. Figure 4 shows a comparison for costs that would be incurred for the number of translated characters in Table 10.

Figure 4: Cost comparison (estimated) between GoURMET and Google based on current BBC GoURMET usage between 7 January and 20 June 2022

The only realistic approach to scale the service up is to introduce additional long-running containers. That is, however, an expensive practice. The only scenario where this isn't an issue is offline batched translation, where low latency isn't essential, and therefore containers can be cold-booted.

As a result, the performance of GoURMET models is limited, and their running costs are fixed

and accrued per second. It also turns billing into a black box, making it hard to analyse the cost-efficiency of the system. In contrast, external providers offer virtually unlimited scalability and charge per translation data used, which is likely much cheaper overall and provides better breakdown and visibility of costs. (For instance, Google models cost 20 dollars per 1 million characters, with the first 500k of the month being free.)

In the future, it would be advisable to ensure that the custom models are multithreaded, utilise a GPU, prioritise short startup times, and have a more efficient and cost-effective deployment architecture.

Nevertheless, the GoURMET models still offer potential advantages as we will go on to explain in more detail in the Conclusion in section 10.

6 BBC Prototypes

6.0 Background

This section describes each of the prototypes developed in the second half of the project. This was originally due to be over a period of 18 months, but this became 24 months following two three-month extensions due to Covid-19.

The words ‘demonstrator’ and ‘prototype’ are sometimes used interchangeably but in the case of GoURMET, there was a notable progression from the early Demonstrator User Interface (which demonstrated the models working) to the later prototypes described below (which tested the models through prototypes).

The translation platform described in D5.3 Initial Integration Report forms the foundation of the prototypes created by and tested within the BBC and DW. These prototypes were built to support further and real-world evaluation of the underlying machine translation (MT) systems.

The original project plan suggested that the models could be tested within two previously built BBC platforms, Alto (which provided an end-to-end video translation pipeline) and Abuja (an MT platform to translate news articles published on BBC websites). At the time it was a reasonable assumption that these would be used, but events and changing priorities in the interim required a revision of the plan.

These included an expression of intent from editorial leaders that they would not be willing to engage with these platforms for future trials. Since the codebase of both Alto and Abuja were ageing, it brought about an opportunity to develop fresh ideas that could be better integrated into the ever evolving BBC tooling and workflows.

These factors corresponded with a transformation in BBC business strategy, as well as changes to the team leading the GoURMET project work for the BBC. At this time the project benefited from a fresh injection of skills that were highly relevant to the tasks involved in the second half of the project.

The new staff members in the BBC News Labs Multilingual Journalism team included one software engineer who had studied translation, another senior software engineer who had studied journalism, and a coordinator who is a long-serving journalist as well as translator. This put journalistic needs and concerns at the heart of all considerations.

The positive injection of new skills and fresh enthusiasm were hampered by diminished innovation appetite among wider production teams due to staffing shortages and work-from-home arrangements amid the global Covid-19 pandemic from early 2020. The project's final year also coincided with the most comprehensive internal restructuring BBC has carried out in decades.

Nevertheless the team delivered on its commitments, and the following tools were developed to deploy and exploit the translation models resulting from the project output.

6.1 Live Pages Translation (LPT)

6.1.1 Background and use case

Live Pages Translation (LPT) is a bold experiment to aggregate all BBC World Service content in one pool and to make it accessible to every staff member across the service by removing the language barriers. As such, it serves the monitoring use case of GoURMET.

BBC News produces output in 43 languages (as already described in section 2.5.3.1). The content follows a broad global agenda with regional and local variances. There is room for utilising machine translation (MT) to cut down duplication of effort across the newsroom for content of a similar or identical nature (e.g. researching and writing the same stories in different languages), as well as amplifying and sharing expertise, and freeing journalists to produce original stories by facilitating reversioning workflows.

Live pages are a recent addition to BBC News digital content, offering a snappy, immediate and responsive content format. Journalists (or a team of them) provide short posts of 1-5 paragraphs on a subject, which can be published quickly. The posts do not require the full, time-consuming treatment of a full article (>500 words) that requires structured writing, formatting and additional 'furniture' (e.g. images and tables highlighting additional facts or statements). As such, live pages are particularly suitable to cover breaking news or fast moving, developing events with multiple aspects or storylines.

6.1.2 Languages

At the time LPT was initially developed, the GoURMET translation models were not yet ready, so for the first project cycle the team carried out trials using all the BBC World Service languages that were available at the time through Google Translate.

While all available languages were provided within the tool, three teams in particular were chosen for a closer examination of team needs and expectations, and to seek feedback on the proposed workflow. The results are described in section 6.1.6.

Following the interim review, Swahili was selected as the first language to be deployed into LPT. This represented the first deployment of one of the GoURMET models in a user facing prototype. As such, it revealed a series of unforeseen issues.

The main issue was that the model was set to work on a sentence-by-sentence basis, which did not reflect the practical usage of submitting larger segments of text. Furthermore, the time the model took to perform the translation meant users would have to wait to receive the results. This was in conflict with the very essence of live pages which is immediacy.

A strategic decision was therefore taken by the team managers to exclude slower models from LPT, and an extensive round of investigations was launched to remedy the situation. This experience highlighted the importance of speed both for the further research phases and for subsequent prototype ideas.

As further GoURMET models became available, Serbian, Tigrinya and Hausa were deployed to this prototype. Swahili was also integrated following the resolution of speed concerns already mentioned, a process which is described in detail in section 8.

6.1.3 Goals

The aim of the LPT prototype was to reveal whether MT in one or more languages is satisfactory enough for journalists to want to use it to create or update live pages on a daily basis.

The initial hypothesis was that if the MT proved to be 'good enough', the next step would be building a pipeline to help journalists in one language service create or update live pages from another language service.

The requirements were that translations must be available on demand, work for a typical Live Pages post of approximately 150 words, that the system would support on average 100 posts sent for translation concurrently, and that page loading time should be less than a few seconds.

Work on the prototype started in May 2020, with the main body of work completed by the end of September 2020. However, the prototype underwent further desnagging and enhancements through to the end of 2020, followed by a trial in the first quarter of 2021.

6.1.4 Architecture info

As already highlighted, with no GoURMET models ready at the time this prototype was initially developed, it was important to have a mechanism that would allow for new models to be added as they became available.

We created a mechanism to allow us to easily switch between different translation models or translation API endpoints for any language (e.g. we might want to use GoURMET models for Serbian↔English and Tigrinya↔English while using Google models for other language pairs).

We wanted to switch between different endpoints seamlessly without code changes because any code deployment could potentially introduce new bugs, and avoiding code updates would make changes quicker.

To solve this we needed to have a one-to-one mapping between language and translation API endpoints, and to store the config in an appropriate place (e.g. Cosmos or S3). It was also important that the config could be defined for the 'test' and 'live' environments independently.

LPT, as a real-time-oriented tool, inherently requires more performant MT models that are able to keep up with large, on-demand translation tasks. The initial architecture of LPT, however, did not account for languages with higher translation latency, which resulted in subpar user experience.

In essence, for a given period, entire article collections would need to be fetched, processed, and translated in full before any data is displayed to the end user. Notably, this worked well in combination with external translation providers, which were able to scale up to such demand.

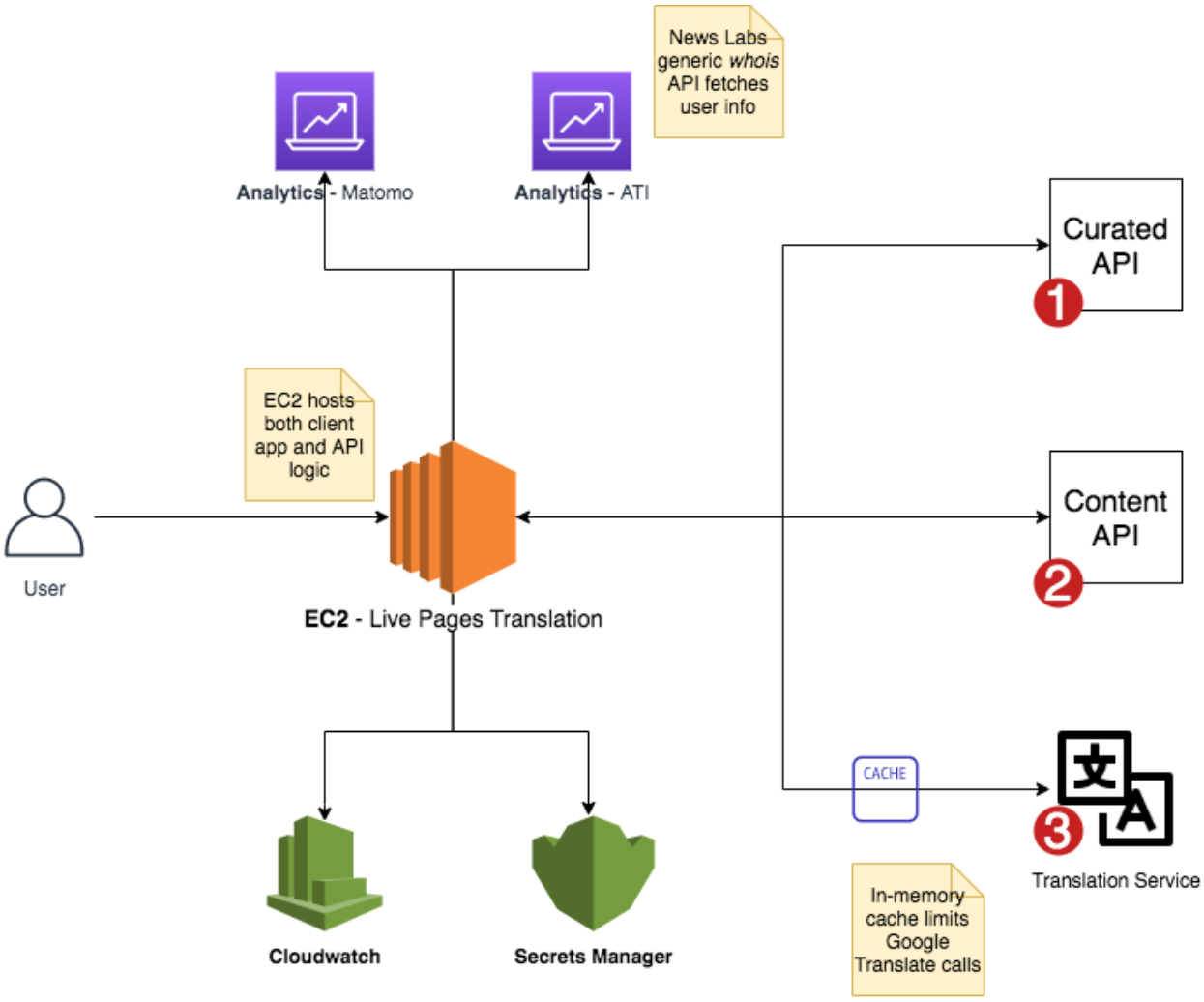

Figure 5: LPT architecture

In order to introduce GoURMET languages in LPT, this expensive upfront processing had to be minimised. This was achieved by moving away from traditional client-server communication and adopting an event-driven architecture utilising WebSockets.

This allowed for greater flexibility in how article collections are processed, especially in terms of translation. As a result, users are able to see individual translated articles as soon as they become available, without waiting for whole collections to be processed.

The final prototype architecture better reflects the real-time monitoring use case and is able to accommodate MT models of varying speeds. Following this change, Hausa, Swahili, Serbian and Tigrinya gourmet models were successfully integrated and deployed into LPT.

6.1.5 Outputs

Figures 6-10 illustrate the key components of a live page, including news updates, correspondent analysis, a summary of key points, related stories, and videos linked to the event.

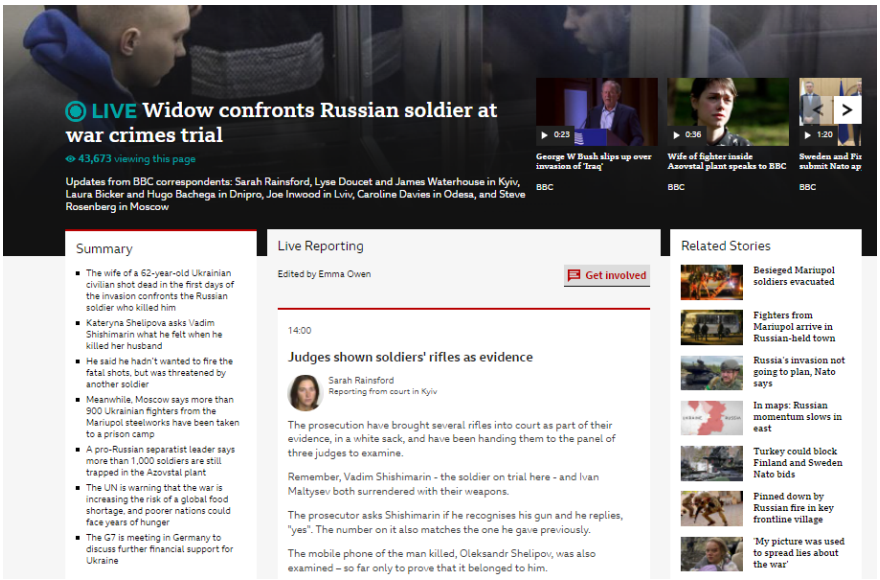

Figure 6: Sample BBC News English live page about Ukraine captured on 19/05/2021 with summary of key events, live reporting stream (middle of the page), key links and multimedia links

The LPT tool brings together multiple outlets, including those illustrated above, particularly during big, shared news events, showcasing the range of treatments and angles available from different services, and on routine days offers a snapshot of the range of stories and events different teams chose to focus on.

The tool aggregates and lists all live page entries in chronological order, regardless of which language they were published in. A simple filter allows a user to narrow down the publishing window (i.e. how recent the posts are) between 1-24 hours, and select a sub-selection of languages and outlets to monitor. The tool enables journalists to see all the content translated into any other language or access the full original post in its original context, and copy the translated content to transfer it elsewhere for use.

The tool aggregates posts in real time from every live page or selected ones, and translates them into any BBC language on demand.

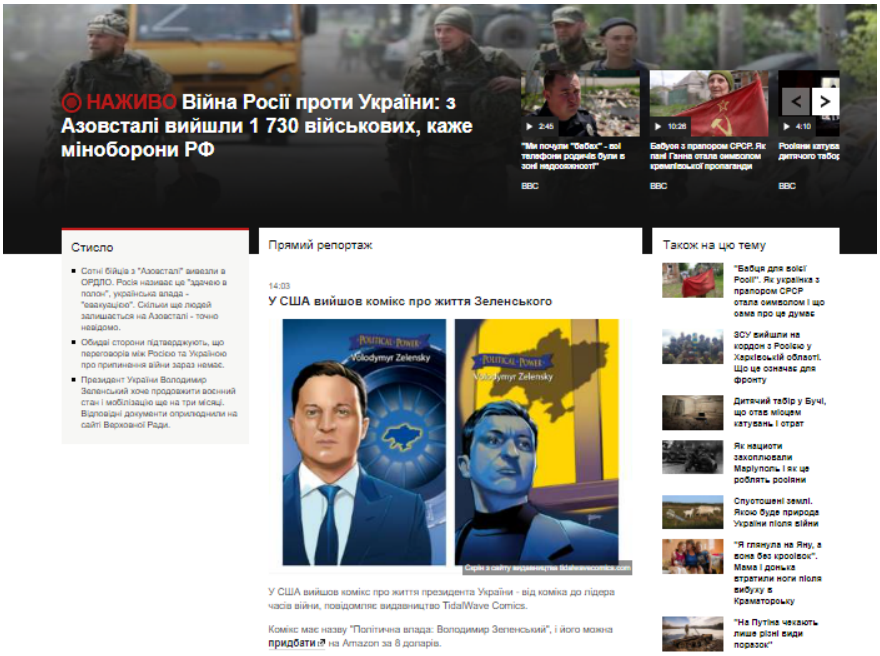

Figure 7: Sample BBC News Ukrainian live page captured on 19/05/2021

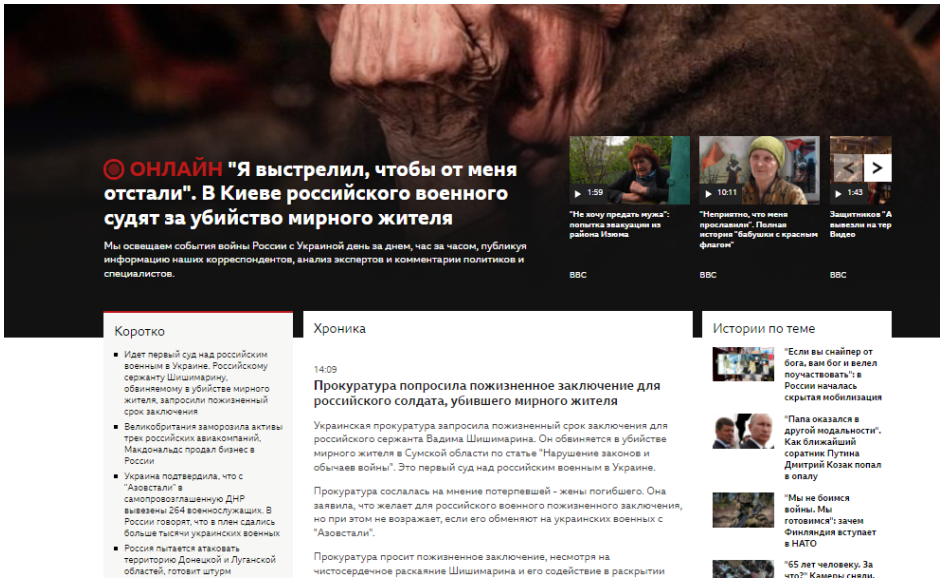

Figure 8: Sample BBC News Russian live page captured on 19/05/2021

The LPT prototype was released in mid-August 2020, with further improvements to enhance the user experience based on feedback. At the time of writing it has been available to BBC journalists as a prototype for over a year and a half and is still live.

It has proven particularly useful during developing events with global repercussions such as the military coup in Myanmar in February 2021, the death of Prince Philip in April 2021, and the Russia-Ukraine War which escalated in February 2022.

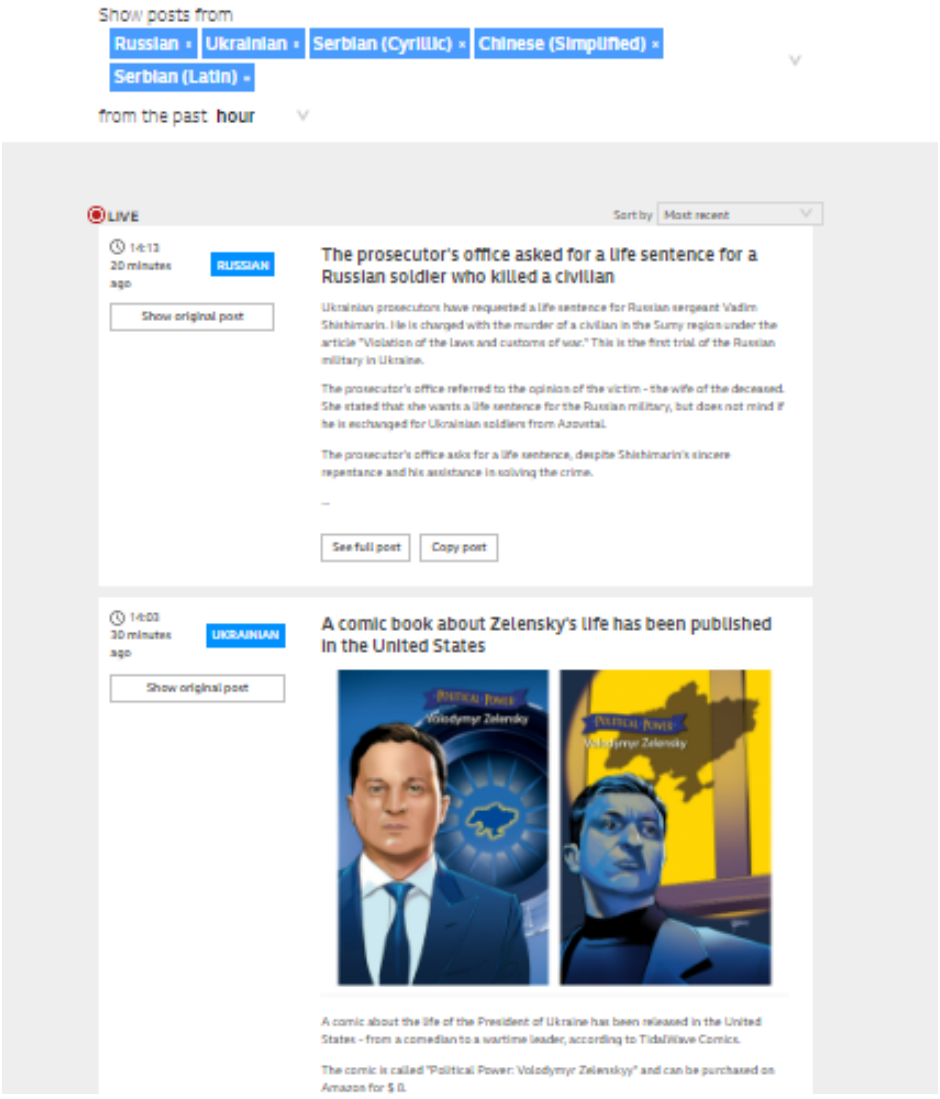

Figure 9: LPT user interface showing a filtered list of stories

6.1.6 Findings from user trials

Figure 11 shows usage of the LPT prototype. You can see there is a big spike in usage from late February 2022 until now which corresponds with the escalation of the Russia-Ukraine War. This highlights the usefulness of MT for monitoring major international events and shows that journalists have the appetite to make use of such a platform in these circumstances.

As well as monitoring usage, we sought detailed feedback from a subset of users as described below:

For the first project cycle in 2021, in the absence of GoURMET models being available for deployment at that time, the team carried out trials with three BBC World Service language teams: Urdu, Hausa and Persian. These teams had similar interests and cultural affinities, and all were interested in content from each other and particularly from the Arabic Service.

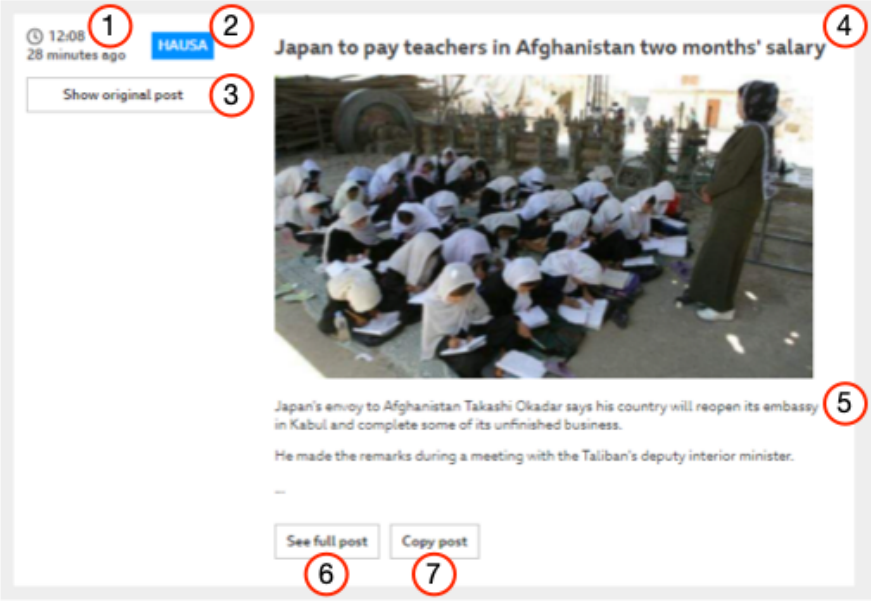

Figure 10: A sample live page post item within the LPT tool showing 1. Publishing time, 2. Language Service, 3. Button to navigate to original post on additional tab, 4. Headline, 5. Top lines of post, 6. Button to expand full post view, and 7. Copy text content on post.

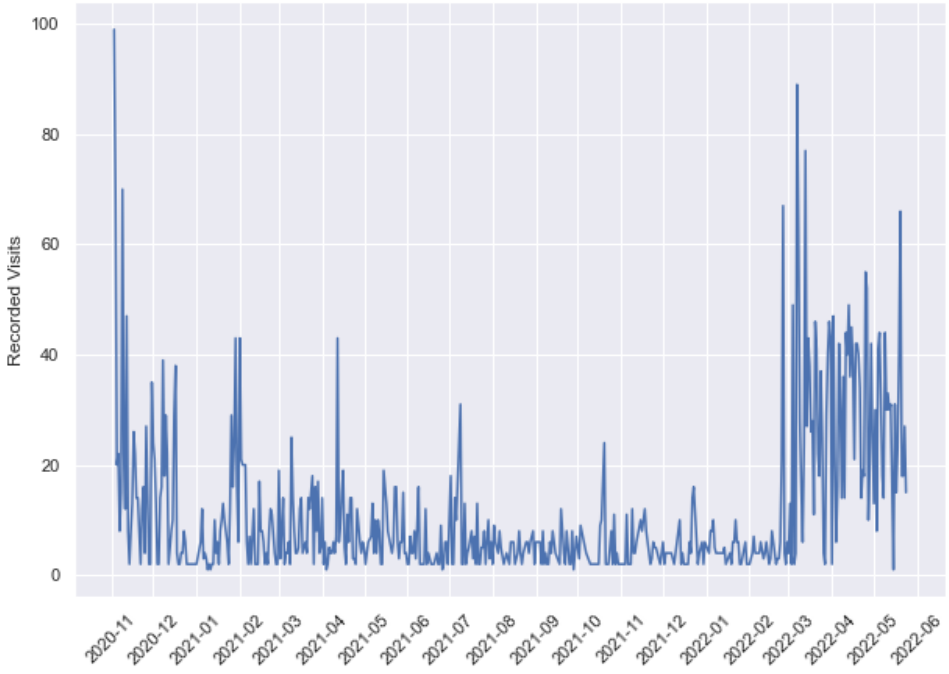

Figure 11: LPT prototype usage

Journalists from each service spent one hour working without translation in their normal workflow, followed by one hour working while being served automatic translations. We found:

- All journalists responded favourably to the idea, agreeing that the automated nature of story discovery from other live pages would make their lives easier.
- All journalists were keen for content to be automatically translated, provided the quality of the translation was good enough.
- Journalists from Urdu and Hausa did not have confidence in the quality of translations into their own language, but were happy to have content translated into English, for them to manually translate it into their language Urdu or Hausa.
- Most journalists asked for an easy way to access and compare original language content. We trialled displaying both the English translation and the original language on one card. Another suggestion was providing a link to view the original language page. This solution was adopted for further iterations.
- Since the English output was enormous in range, with several Live Pages running in tandem across UK nations and regions, when English was selected in the dropdown among languages to follow, it risked drowning out most other teams' content due to sheer volume. Therefore, English-language Live Pages were split up based on their URL directories to be filtered out as necessary under English International, English Local and English National settings, with the assumption that most global stories would be under English International.

Building on these results, a further iteration of the LPT prototype was developed which was then further tested. The key findings of this second round were:

- Topic filtering was suggested during the trials as a 'priority' feature during the trials, and should therefore be added to any further iterations (although our focus moved on to the second prototype, Frank, described in the next section).
- Covid-19 measures severely limited team availability to try out the tool, as well as reducing the time teams devote to monitoring others' content for enrichment purposes due to reduced headcounts, repurposed workflows and issues with connectivity.
- The deployment challenges with the models were more involved than we had expected. In particular addressing issues with speed turned out to be a major challenge (as speed was not considered the top requirement by the research partners but turned out to be a significant issue for test users due to the nature and immediacy of the content).
- Quality was of utmost concern for the users. Since live pages are highly dynamic, developing and rapidly changing, translations need to be highly accurate to be useful to anyone or they risk amplifying incorrect information. The tight turnover of the live pages (i.e. publishing a brand new post every 20-40 mins with a one or two person team), also left little space for editorial safety nets to be triggered and could lead to misleading results.

An example the Turkish service provided was from the time the French President Emmanuel Macron suffered from coronavirus (see Figure 12). The early version of the GoURMET Turkish model (v1) did not deal competently with the new vocabulary around Covid-19 and the translated story turned out to be on how Macron suffered from coronary disease, and also signed an accord at his coronation ceremony. We explain how we worked collaboratively to resolve issues like these in section 8.

Figure 12: The Hausa>Turkish translation of a story from 17 Dec 2020 about Emmanuel Macron catching Covid-19. Since the word coronavirus could not be translated appropriately, the headline reads: 'French President Emmanuel Macron signed an accord at his coronation ceremony.' The first sentence follows to add Macron has 'caught coronary heart disease' and '...that he will remain locked for seven days and continue working'.

In conclusion, LPT is a live prototype in active widespread use across the BBC with considerable further potential to help journalists monitor local source material, and use this material to enrich and update their content.

The Russia-Ukraine War, which escalated with Russia's invasion of Ukraine in February 2022, has been marked with a clear boost in usage of LPT, growing more than threefold, with outlets broadcasting in English turning to WS Language outlets to follow the immediate coverage from the ground. This is an organic boost, potentially through peer-to-peer recommendations with no formal promotion work conducted on the part of News Labs.

One user commented: *I think there are two really helpful things about the LPT. First, coalescing multiple live pages into one single area. That's really helpful – it's much easier than keeping multiple browser windows on my laptop. For example, I'm able to monitor the Ukrainian and Russian live pages simultaneously in just one place. The second thing is that the non-English*

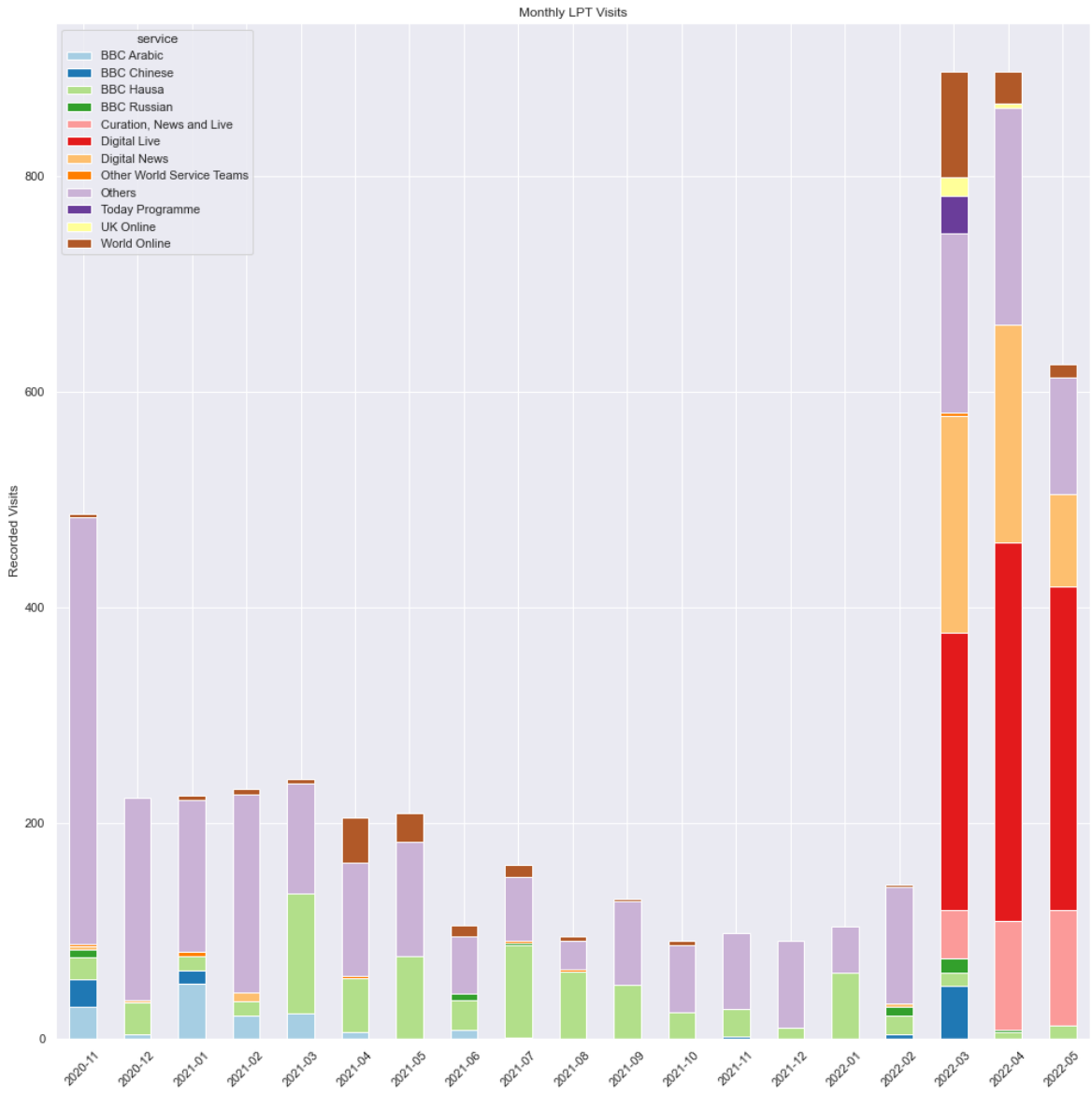

Figure 13: LPT usage by team Oct 2021 to May 2022. The spike in spring 2022 corresponds with the escalation in the Russia-Ukraine War.

live pages themselves are a very valuable resource – all the more so when they are automatically translated. Reason being that they pick up stories from interesting local media sources which I don't follow (lots of Ukrainian and Russian news agencies).

Another added: *The tool is extremely useful for monitoring our Ukrainian and Russian service colleagues' live pages. The 'show original post' feature is helpful, as the teams often include links to the source in the actual post. We usually check sourcing ourselves and re-write material, but it's very useful for alerting us to content we might not be aware of.*

LPT requires better search and filtering facilities to rapidly direct users to the content they deem relevant. As a result of this finding, this requirement was developed and deployed in the subsequent prototype. It would be advisable to revisit LPT to retrofit these capabilities if and when there are further plans to formalise the exploitation of the tool.

It became clear at the end of this trial that BBC editorial management would only approve use of models that would not have a negative impact on the user experience, which in turn led to the collaborative improvement work described in section 8.

One key learning, during and after the trials, was the extent of users' willingness to work with imperfect machine translations. Journalists from several teams explained that they didn't have time to verify data from other teams, or to fix translation errors, while both the originating and recipient teams are under pressure to deliver a fast moving story.

When asked what kind of content would warrant the investment in time for editorial verifications, the responses converged around:

- Exclusive or original stories
- Stories showcasing expertise
- Stories without publishing pressure
- Stories that could meet the needs of 'underserved' audiences

Live pages offer an alternative format for presenting news to audiences, but only the teams with the largest staffing levels are in a position to continuously maintain such an offer. It was a good place to begin our experiments, but as we were keen that the language tools saw wider use, the logical next step was to develop a prototype that covered the main body of production (i.e. news articles). This would be our second prototype:

6.2 Frank (Lingua Franca)

Frank is the short form name we gave to our 'Lingua Franca' prototype, referring in this case to English as the shared language of almost all BBC producers regardless of their outlets. This prototype was conceived as a BBC World Service content discovery tool, aimed at congregating the main body of digital content BBC teams provide on an equal footing, removing the language barriers between them with a view to ensure:

- Transparency across the range of news content and hence to support editorial compliance and commissioning processes.
- Original content with robust and proven performance can be identified and showcased to other language services to have onward journeys to amplify impact, boost the sense of distinct content offer and to serve key audience need areas.

6.2.1 Background and use case

In light of the internal restructure of the BBC News Labs Multilingual Journalism team coordinating the GoURMET project in autumn 2020, and informed by the findings of the LPT trials, we conducted a survey to identify the most up-to-date needs of BBC World Service teams, to explore where further opportunities for multilingual solutions could lie, and to discover how journalists perceived potential benefits of machine translation (MT).

Although an increasing number of journalists reported they were open to using MT tools for monitoring, there was still palpable resistance to the content creation use case due to concerns around accuracy, style and overall reputational risks.

When asked about what might make MT a worthwhile route to take, journalists often stated they were keen to try out the new technology, with several pointing out that targeting a slower news circuit seemed more amenable for building a safety net to ensure accuracy, quality and value around MT output. Journalists wanted time to be able to check the accuracy of translated information and refine the style of content.

At the same time, a significant subsection of users from several different teams suggested filtering out 'routine, daily news stories' that report on straight facts of an event, saying often these stories were already coming from sources in English and that journalists could reversion these more quickly if they simply wrote them from scratch.

What they were most interested in were the human-centred, personal and/or unusual stories from around the world that people would like to read but would not be able to find anywhere else, as in this case MT could help them ensure they could address any content gaps and have more or less equal volumes of stories serving every audience segment.

During the last decade, BBC World Service has been classifying content under six categories corresponding to audience 'News Needs' (i.e. what audiences want from the content). These are:

- Update me
- Educate me
- Give me perspective
- Inspire Me
- Divert me
- Keep me on trend

The interviews conducted around the time of the survey, and since, have consistently demonstrated that journalists (with the exception of Hindi team) did not want to see **update Me** stories, due to the fact that these need to be highly accurate and relevant to audience.

Neither did they want to see **keep me on trend** stories from other language teams, since they expected these stories, often involving viral clips or events heavily reflecting shared local knowledge or culture, would by their very nature only be of interest to a certain country or region's audience.

Our original proposal was to limit the Frank prototype to only show stories under the **inspire me** and **give me perspective** categories. However, more categories, four in total, were added to the prototype, with an additional feature allowing users to filter stories by the news need(s) they are interested in.

This survey also reinforced what we had been hearing anecdotally – that the BBC World Service lacked a complete overview of the range of content being produced day in day out.

It followed that:

- Teams on the non-English outlets lacked the means to highlight their original, special and successful content and to share it with other teams, or colleagues within their own team, in order to amplify usage.
- Teams did not have knowledge across languages as to which services have produced content that adds unique local perspectives to global stories.
- BBC World Service leaders were unable to track which language services reversioned stories that had been distributed centrally, and were therefore unable to monitor the usefulness of their selection and offer. This was primarily due to the language barriers between teams (i.e journalists or hub editors speaking a limited range of the languages they oversee).

Using MT could help bridge these gaps and address the points above while achieving goals detailed further below.

6.2.2 Languages

For Frank we envisaged a bidirectional language flow, with a heavy emphasis on translations into English which were generally deemed more robust. As its name suggests, it first and foremost transferred the entire portfolio of the BBC World Service offer, estimated to be around 400 articles per day, into English, creating a level playing field for further analysis and utilisation.

Work on the prototype began in July 2021 as the first two sets of GoURMET translation models were coming online. As we explained in section 2.5.3.1, after an initial period of examination the team reverted to utilising Google models having previously considered a workflow that supports GoURMET models with Amazon Translate languages.

While the tool served all BBC World Service languages, a subset of languages which appeared to provide adequate translations based on a small sample survey conducted in June-July 2021 was selected for outreach to teams to engage them for the user trials, including potential post-editing.

Eventually seven GoURMET languages (each language paired with English) were deployed on this tool. These were:

1. Swahili
5. Tamil
6. Serbian
12. Tigrinya
13. Pashto
16. Urdu
17. Turkish v2

6.2.3 Goals

Frank was built to serve complementary aims of GoURMET and BBC World Service growth. Its strategic goals were to:

- Highlight the best of BBC's global journalism and enable the best journalism to travel further
- Maximise impact for audiences for international original journalism, investigations and global events
- Help scale down duplication
- Speed up translation processes and free up journalists' time
- Help maintain the BBC's relevance with regard to fast-developing MT technologies

It would do this by delivering the following key results:

- **Pinpoint and amplify** content that could be regarded as **original/field journalism**, or demonstrating shared themes or values, and as such has potential to be offered to other teams.
- **Mapping affinities** between neighbouring teams for **'desire pathways'** for reversioning and seeking where there are opportunities to signpost the unique perspectives to global stories.
- Offering hub editors, commissioning and planning editors **a full and immediate overview** for **editorial compliance**, involving what each team is producing and boosting co-production opportunities.
- **Exposing journalists to an MT-assisted workflow** where they could view multiple original versions side by side, offering reassurance regarding doubts about accuracy.
- **Providing a platform** where journalists could conduct **post-edits** in order to collate more data on how quick or useful the models have been with a view to build future retraining cycles based on feedback.
- **Informing next steps** on translation projects downstream, which might involve:
  - **A monitoring tool** that aligns, classifies and filters out content on whether it is reversioned, original or general content.
  - **Better onward journeys** by suggesting to journalists the kind of content that could be right for underserved audience segments
  - **Audience-facing enhancements** such as a semi-automated view of global perspectives on a particular story/theme in a newsletter or notifications.

6.2.4 Architecture info

Frank required a language agnostic API to identify reversioned and original content across the BBC that could be marked for potential reversioning across language services.

It then required a human readable interface to show the output of the API and allow us to gather editorial feedback about the quality of content classification.

Figure 14 shows a conceptual overview of how translated content is showcased to users.

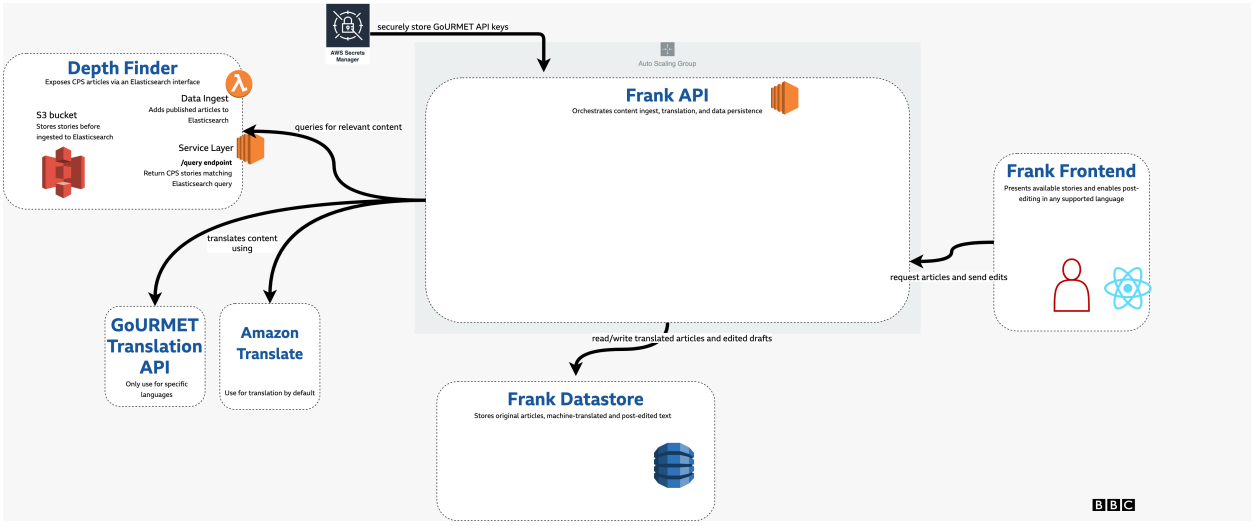

Figure 14: Frank conceptual overview

Users then have the option to make edits to a translation, and to translate the full article into other languages (see Figure 15).

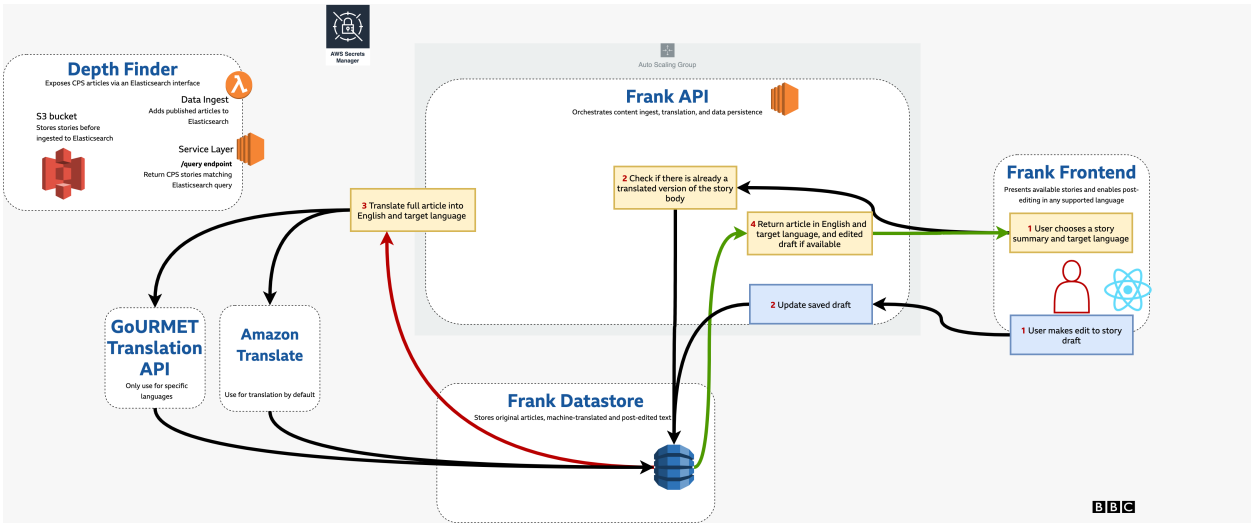

Figure 15: Frank conceptual overview with further stages

Unlike the LPT prototype, Frank performs translations in the background and saves these in DynamoDB, a fully managed proprietary NoSQL database service which is part of the Amazon Web Services portfolio. This gets around many of the issues with speed faced by LPT.

As an example, a BBC News article bbc.com/persian/magazine-57612202 is translated from Persian to English and stored in JSON key-value pairs that include ratings and analytics:


```
{
  "id": "57612202",
  "articleType": "ws",
  "analysis": {
    "newsNeed": [
      "Give me perspective"
    ]
  },
  "publishedTime": "2021-07-14T08:43:09+00:00",
  "image": {
    "href": "http://c.files.bbci.co.uk/4F60/production/_118502302_gettyimages-504845290.jpg",
    "altText": "The results add pressure on Beijing to boost measures for couples to have more
        babies"
  },
  "content": {
    "source": {
      "language": "fa",
      "headline": "سرشماری چین: داده ها کندترین رشد جمعیت را در چند دهه اخیر نشان می دهد",
      "summary": "نتایج به پکن فشار می آورد تا اقدامات لازم برای بچه دار شدن بیشتر زوج ها را افزایش دهد"
    },
    "en": {
      "translationModel": "aws-fa-en",
      "language": "en",
      "headline": "China census: Data shows slowest population growth in decades",
      "summary": "The results add pressure on Beijing to boost measures for couples to have more
          babies.",
      "rating": [
        -2,
        -1,
        0,
        1,
        2
      ]
    },
  },
  "analytics": {
    "engaged": 43,
    "vistis": 3372,
    "views": 6732
  },
}
```


6.2.5 Outputs

The Frank tool provides two key views:

- Dashboard
- Editor

The first dashboard view initially just aggregated language service stories: Frank automatically aggregates articles from all language services provided they have been tagged with the four news needs described earlier at the point of publication in the BBC's main content management system CPS.

Work is now underway to ensure Frank can also pull in articles from other and any future content management systems through a translation infrastructure pipeline that has been developed in view of the learnings from the earlier LPT and more recent Frank trials.

Further work in March 2022 added a new dashboard view, presented via an additional navigational tab. This new view was requested by the Digihub team, part of the BBC World Service that creates content in English for distribution in other languages (i.e. across the range of BBC World Service language services).

As of summer 2022, Frank now has four navigational tabs (see Figure 16):

1. Frank – a landing page to display a combination of all stories
2. WS Stories – language service stories
3. Digihub Stories – stories created in English for eventual distribution in other languages
4. My Bookmarks – an individual selection of stories a user can bookmark for later

Figure 16: Frank prototype tabs

These tabs display content as follows:

6.2.5.1 Frank tab

The Frank homepage is a dashboard conceived to offer a combined view of the WS Stories (language service stories) and Digihub Stories (stories from the Digihub team, createad in English for distribution in across other language services) in the same space.

6.2.5.2 WS Stories tab

The WS Stories dashboard displays stories published in the last seven days translated from the full range of BBC World Service languages into English using a combination of GoURMET models and Google Translate. The dashboard usually features around 200 stories, although the number fluctuates depending on the publication traffic from teams on any given day.

In all cases, for each story, the dashboard includes a headline, image and summary, and data about which language service published it, when they published it, a word count, how many page views it had, and how much time each user on average spent on the page (engagement time). The last of these is increasingly used by editorial leaders as a means to signify value to the audience.

The dashboard features a filter to help journalists drill down to the kind of stories that serve a particular user need (one or a combination of the four news needs within scope) or to only view stories from specific language services.

Stories can be sorted and displayed either in chronological order (newest on top), by number of page views (highest on top), or in terms of engagement time (longest on top).

There is also an extensive search function, which filters content based on whether the search term appears in the headline, summary or main body (with the implied assumption that the search result

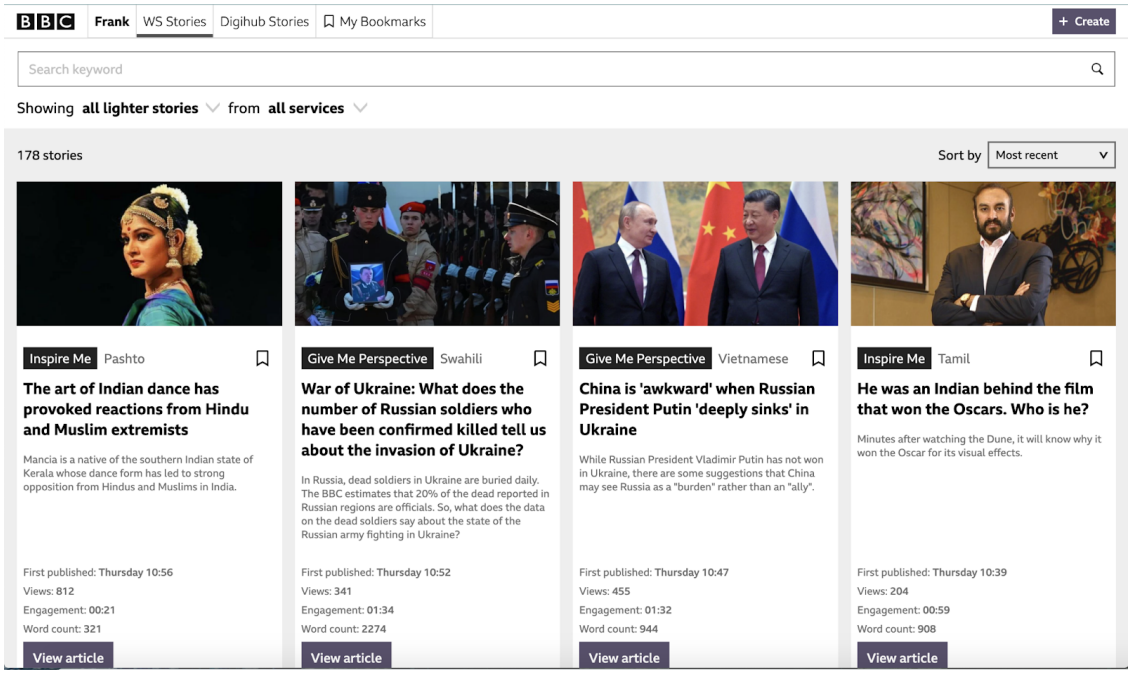

Figure 17: Frank prototype WS Stories dashboard view

is more likely to be relevant to the search term if it appears in the headline or summary) and a bookmark feature to allow users to go back to stories they find of interest.

Each of these features provide journalists with options to discover the kind of stories that would be most relevant to their needs with ease.

For instance, if a journalist wants to find out which lighter stories about women were received most attentively by the global audiences (longest engagement time to read the article through), a simple search would offer the result shown in Figure 18.

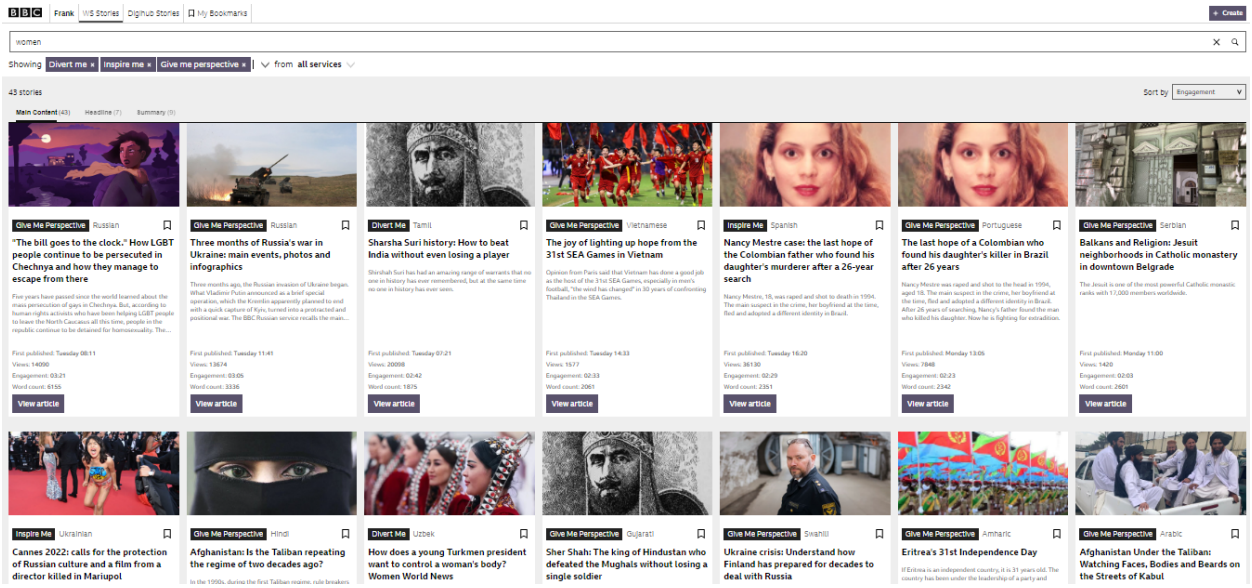

Figure 18: Frank search results for investigative, inspiring and educating stories involving 'women' with longest engagement (accessed 25 May 2022)

Figure 18 illustrates the benefits and insights of the discovery function of Frank:

A story about a father's quest to find his daughter's murderer after a 26-year search (5th and 6th cards from the left on the top row) was first published by the Portuguese team. This was then picked up and reversioned by the Spanish (Mundo) team, where it attracted five times more page views (7848 vs 36130). In both cases, users have stayed on the page to read the story through (2'23" to 2'29" respectively).

This goes to show that enabling teams to identify and reuse relevant content from each other could be a valuable investment for both parties, amplifying value and reach for the BBC, by generating secondary audiences much larger than the intended primary audience.

Changing the filter and display settings to search for stories with the most page views about women (simply using the search term 'women') offers slightly different results, allowing journalists to make a different set of choices as shown in Figure 19.

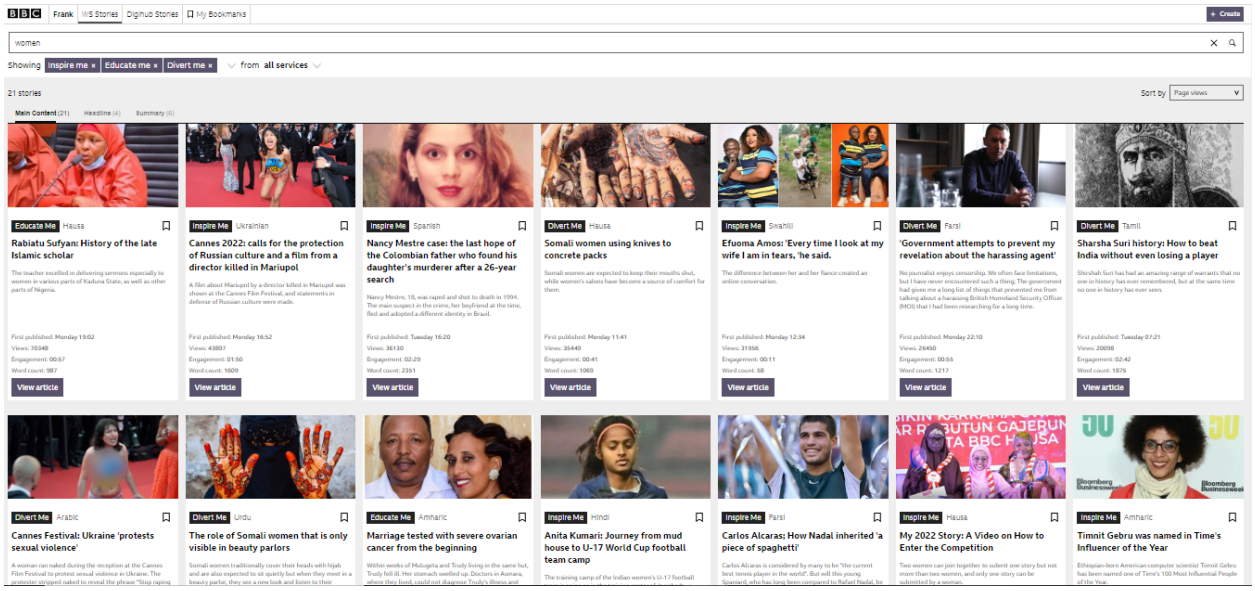

Figure 19: Frank search results for inspiring, diverting and educating stories about 'women' with most page views (accessed 25 May 2022)

6.2.5.3 DigiHub tab

The BBC World Service has a Central Services team that manually selects, processes and redistributes content across TV and digital outlets. This team has been a key stakeholder for GoURMET, particularly during the second half of the project, and in the later stages of the development of the Frank prototype.

The Central Services team managers were very enthusiastic about the discovery capabilities Frank could offer. However, they wanted their digital content in particular, which is curated by a subdivision of their team called DigiHub, to appear alongside, and preferably more prominently than WS Stories in Frank.

Despite their ten-year history, the DigiHub team had no dedicated publishing space available, and distribution of the content (in English for reversioning into target langauges) was still via emails.

Therefore, work was conducted between January-March 2022 to offer the DigiHub team a means to manually ingest their stories into Frank where they could start their onward journeys. This would give the team a much sought-after feature of being able to track where it has gone on to next.

The DigiHub tab, created in accordance with these requirements, aggregates BBC World Service stories that have been manually ingested by the DigiHub team. These can then be translated into target languages using Frank's interface.

Articles are only kept for five days and then purged to ensure the stories on offer are not stale or out of date. The time left before expiry is displayed on each card. It is also possible to pin 'evergreen' stories with longer shelf life to the dashboard.

In addition to the DigiHub tab, which includes all the features already described on the WS Stories tab, all DigiHub articles have three statuses reflecting workflow needs:

Ready – Article is ready for use.

Updated – Article has been amended/updated since it was first submitted to Frank. Reasons for the update will be displayed on the dashboard.

Embargoed – Article should **only** be published on live sites **after** the specified embargo time expires. (Journalists can start working on the content, but cannot publish).

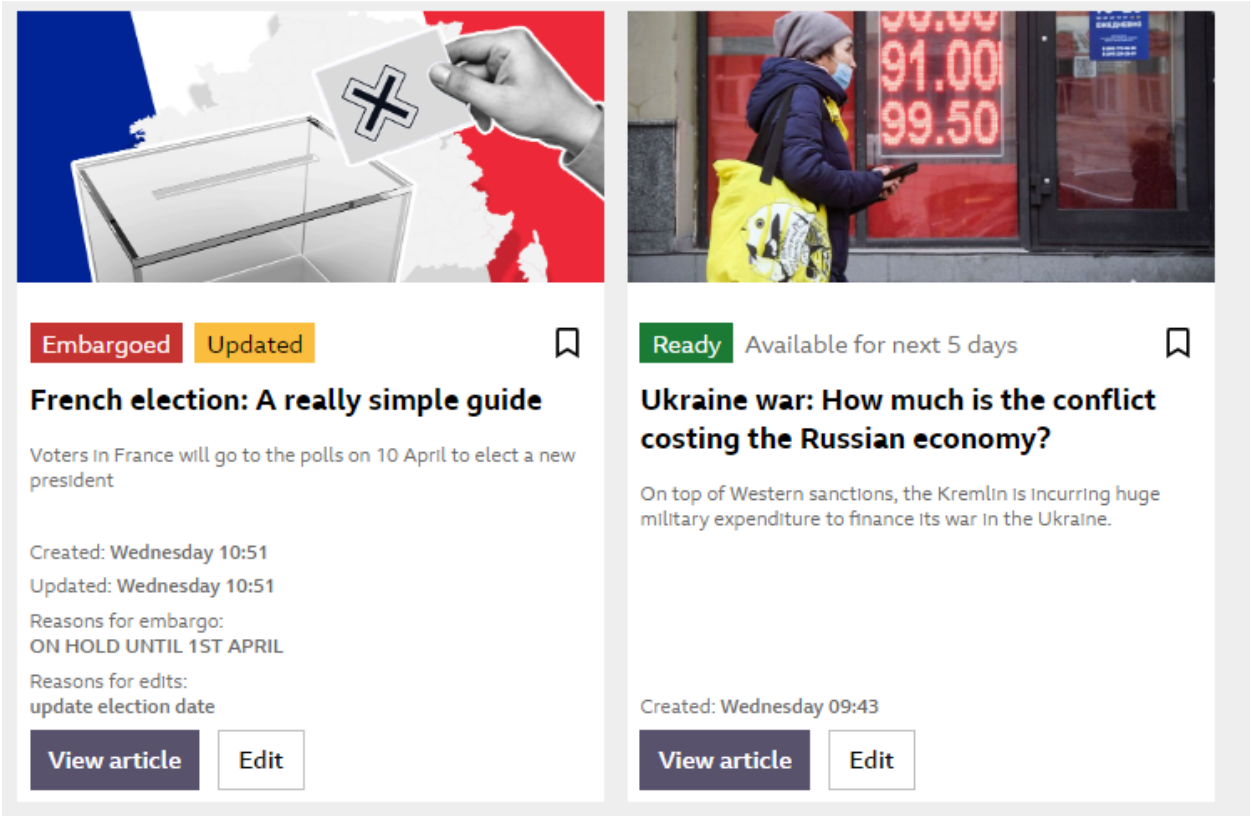

Figure 20: Frank Digihub stories

There are plans to run a full DigiHub trial on Frank, which has been put on hold due to severe staff shortages in the team. The DigiHub team also has further suggestions for additional features to improve workflows.

The eventual goal is to have the full endorsement of Central Services to drive the bulk of the digital reversioning traffic into Frank and to clear the way to transform the prototype into a fully maintained product.

6.2.5.4 My Bookmarks tab

This tab displays stories that have been bookmarked for ease of access.

6.2.5.5 Editor view

Having selected a story via one of the tabs described above, a user will be taken into the Editor view.

Frank embraces the multilingual skills of journalists by offering them side-by-side translations which can be edited seamlessly. Clicking on the View Article button any story card takes users to this secondary layer – the Frank editor view.

The left hand side of the window is the 'view' window, offering the English translation, the original text and the first iteration of the machine translation into the selected target language on a tabbed display.

Figure 21 shows the original story in Serbian, its translation into English (as it was displayed in the dashboard view) and its translation into the selected target language, in this case Turkish.

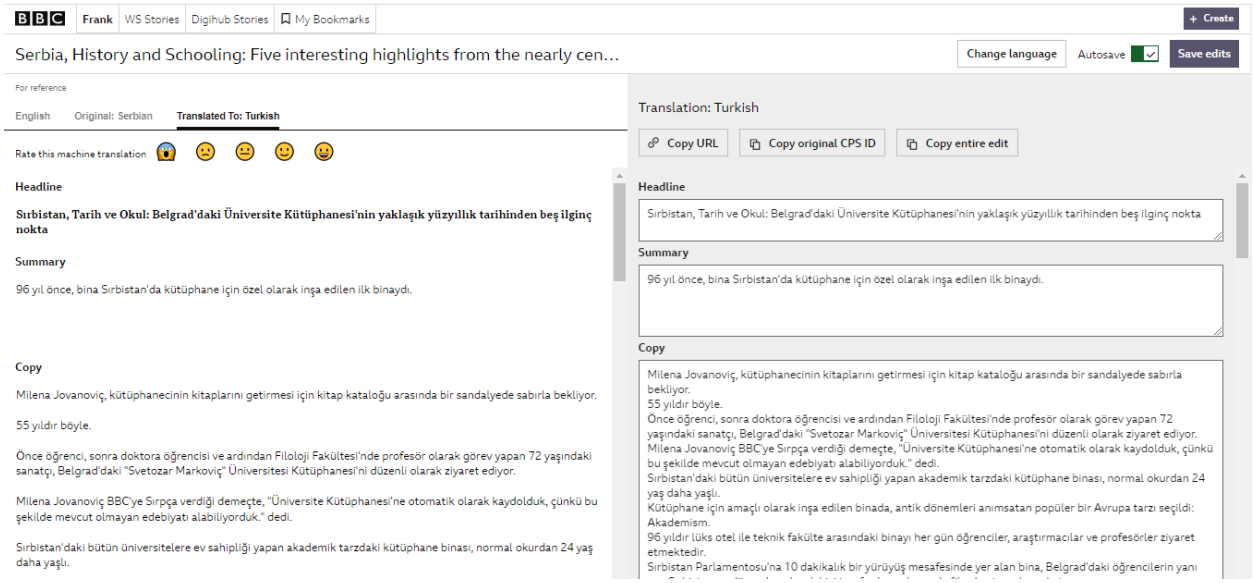

Figure 21: Frank prototype editor view

The design of the Frank editor view empowers journalists who may be speaking multiple languages across potentially closely-related teams (e.g. Serbian/Russian, Urdu/Arabic, Pashto/Persian etc) to compare translations and arrive at the best possible outcome by viewing the original source and its English translation along with the target translation. The left hand side also offers users a quick means of rating the translation through emojis.

The right hand window is the ‘edit’ space, where the journalist can make changes to the target or English translation, which is then saved and displayed for the next user who accesses the same story. This potentially enables originators of an article to offer a ‘validated’ translation in English that can serve as a master for further translations downstream.

A full validation workflow that will allow for both proactive and responsive/on demand validated translations to be featured more prominently on the main dashboard is among ideas considered for further iterations of the tool.

The tool allows journalists to copy the CPS IDs (content management system unique identifiers) for cloning body architecture and features such as images, links into their own workspaces, as well as copying the finished translation into this closed space manually, due to the current lack of an automated back-end door into the publishing tool. A future aspiration for a productionised workflow involves fully integrating Frank into the CMS systems for publication.

6.2.6 Findings from user trials

Frank was launched in August 2021 and its usage was recorded. Every time a user goes to the Frank dashboard, the visit is logged.

Figure 22 shows the total log entries for each day between September 2021 and May 2022. If people move between the pages in Frank, this is recorded as multiple entries.

According to the data, Frank has been used consistently since it was first made available with some peaks of activity from time to time. There was an initial peak when Frank was first launched and offered to staff and while it was being trialled more heavily.

It is encouraging that the prototype has seen continued use in the absence of internal promotion and formal endorsement by editorial management for its day to day use, suggesting that those who were made aware of it initially are still making use of it.

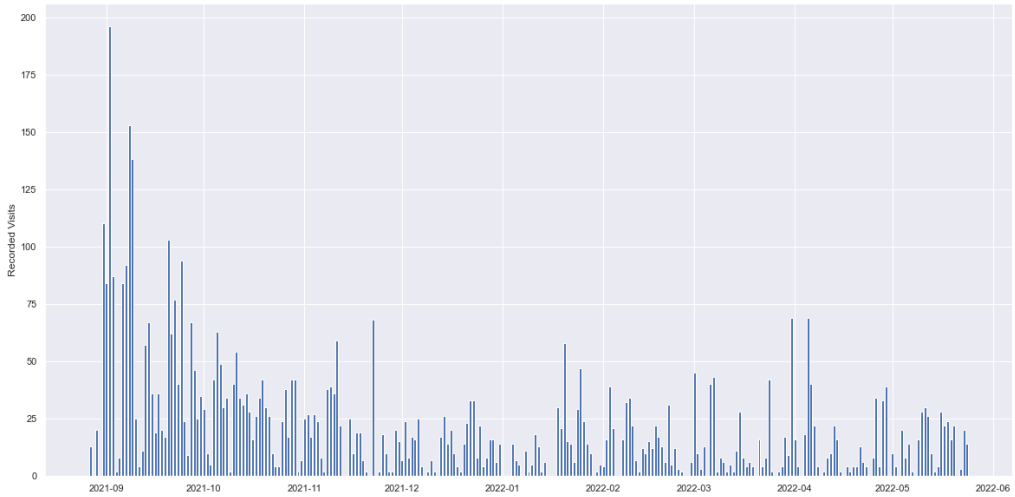

Figure 22: Frank prototype usage

We approached some of the teams that had been using Frank to take part in trials during the period of September-October 2021. We sought feedback both from teams that had been using GoURMET languages and non-GoURMET languages. Teams were selected with input from The BBC World Service Growth Unit, who are tasked with improving performance, to ensure we did not overload teams who are already involved in other projects.

Nominated journalists from the Arabic, Chinese, **Hausa**, **Igbo**, Russian, **Serbian**, **Tamil** and Digihub teams were approached (GoURMET languages in **bold**). All of them had been asked to use Frank in their own time and translate stories into and out of English and share their views at the end of a month-long process.

The trial aimed to separate the **experience** of the tool and discovery processes from the **quality** of translations. The overall feedback was positive. Every interviewee said they would recommend the tool to a colleague. Here is some of the feedback:

6.2.6.1 Tool as a 'marketplace' of content and discovery

Frank has proven to be a success for verifying the underlying idea about the benefits of boosting transparency across teams:

> *Before Frank I was visiting [individual World Service] websites to see what they are covering but now it's easy, I know good stories are coming to Frank.*
>
> *It helps with finding relatable stories from Somali, Swahili, Arabic, Hindi, Turkish, Persian – all these services have stories we have connection to, have cultural affinities, like women's issues; they also work well with our audiences.*
>
> *It gives a very good view of what stories are interesting, what we might have missed.*

The tool still has untapped potential, with an increasing number of articles being showcased (see Table 11).

A similar upwards trend is also visible in weekly figures, with showcased article views climbing from double digits to triple digits in November 2021, and staying above 600 per week since April 2022.

As a further step to facilitate the workflows, the BBC's CMS product team has generated an 'original journalism' tag. However, the roll-out of this to production teams has been held up by editorial discussions about what to include.

6.2.6.2 Tool as a means to speed up the reversioning workflow

When asked whether editing a machine translated story is faster than writing from scratch, the answers varied considerably depending on how well resourced the languages in question are, and the kind of story/domain involved:

> *[Time required] . . . depends on the story, if the story is really sensitive I have to verify all the information – this may take longer. For less risky stories...it will take one day. Whereas writing a story in this genre from scratch takes me about two days.*

Table 11: Number of articles displayed in Frank by month

Month	Articles displayed
Aug 2021	76
Sep 2021	688
Oct 2021	395
Nov 2021	351
Dec 2021	564
Jan 2022	586
Feb 2022	609
Mar 2022	591
Apr 2022	2729
May 2022	2409

I choose Frank's content if there is not sustaining material available – I prefer it over the BBC News website as it is faster to rework than to write from scratch.

[Frank] increases the speed of reversions, translations, some can take two hours or so if they are longer. It is three times faster now; I need time to check translations after but it definitely saves my time, it has a big, wide offer of topics.

Speed of versions depend on the type of story. For example when doing a story from Mundo on Nuremberg trials, it took three hours but if I was doing it from scratch it would take a lot longer. But for a shorter straight story from English – I'd rather do it from scratch as it is faster.

6.2.6.3 Quality of translations

Most of the journalists who tried the tool, all of whom are fluent in other languages and most in multiple languages, said the translations were better when translating from another language into English, rather than from English into another language. None of the journalists said the translations were better from English into another language. This confirms our understanding from data driven evaluation that translation into English nearly always performs better than translation from English (e.g. see D5.6 Evaluation Report section 4.1 Summary of Results).

The evaluators often reported that the quality was adequate for accounts consisting of short, simple sentences. However, quality often appeared to deteriorate with longer, more complicated sentences involving conjunctives. Issues were more evident and frequent in least resourced languages (Amharic, Yoruba and Tigrinya had seen challenges) with some journalists confiding in person after conducting the evaluations that they *'could not know to what extent (they) could rely on the translation'*. Below is some more of the feedback provided at the trials for Frank:

I don't have any problems with English translation – 95% accurate; Hausa has problems with idioms; overall score is 7 over 10.

> *[Quality] depends on which languages it's from, some major languages (Spanish or French) are pretty good. (6 or 7 over 10). It is not necessarily good in less common languages (e.g. Kinyarwanda, Gujarati).*
>
> *In terms of the language, the machine is a machine, it's not human – it lacks the warmth and colour – it's not like the way people talk. We need content to be more pedantic and aesthetically beautiful.*

One sample from the Russian service related to a failed assasination attempt on an associate of Ukrainian President Volodymyr Zelensky, where shots were fired. Reading the story from the Ukrainian>English translation, it appeared as if the attack had killed the intended target, whereas the original explained that the official got away with minor injuries.

This last example illustrates the biggest challenge the work has faced. Editorial leaders regard MT tools as a route to amplify mistakes or substandard journalism, through either MT-related errors or failures to detect editorial shortcomings in the original content while focusing on the translation. Thus, for many risk-averse journalists, MT represents a liability that needs to be carefully managed if it is to become an opportunity to be embraced.

This is the reason BBC News Labs has been asked to not extend trials further, until verification solutions can be developed. We were also asked to add disclaimers (see Figure 23) on the Frank landing pages, making it very visible to users that the content is derived from MTs and as such needs to be carefully considered before it is used.

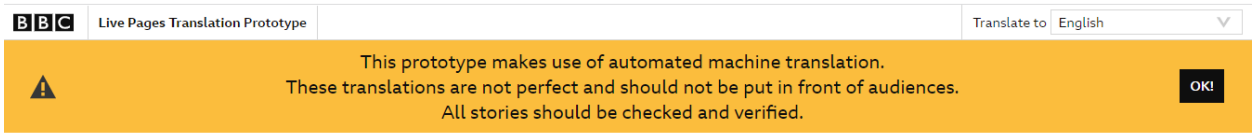

Figure 23: An MT disclaimer appearing on all Frank landing pages

6.2.6.4 Lessons learned

The trials have confirmed that the hypothesis of moving all content into English using MT yields very positive results. Serbian, developed under GoURMET, is potentially one of the better performing translation models, which the team quickly benefited from as they embraced the MT workflow. The team have also borne out our hypothesis that high quality correlates with perceived usefulness of the tool by registering habitual use many months after the initial trial.

Despite a sharp drop following the end of the trial, during which time additional resources had been secured to monitor the tool for content of interest, around a fifth of users continued to come back to it. Two of the teams which continue to use the tool, Urdu and Swahili, are under the scope of GoURMET. It is worth noting that a second sharper reduction happened around the time that the Russia-Ukraine War escalated. This suggests that as teams consolidated their workforce to boost the 'must have' coverage on war, the 'nice to have' sustaining content from Frank became less useful or appealing.

Our experience with Frank has shown us there is certainly an opportunity space and a degree of interest to try out new solutions involving MT. The technology is seen to be ripe for monitoring in particular, but not for content production yet.

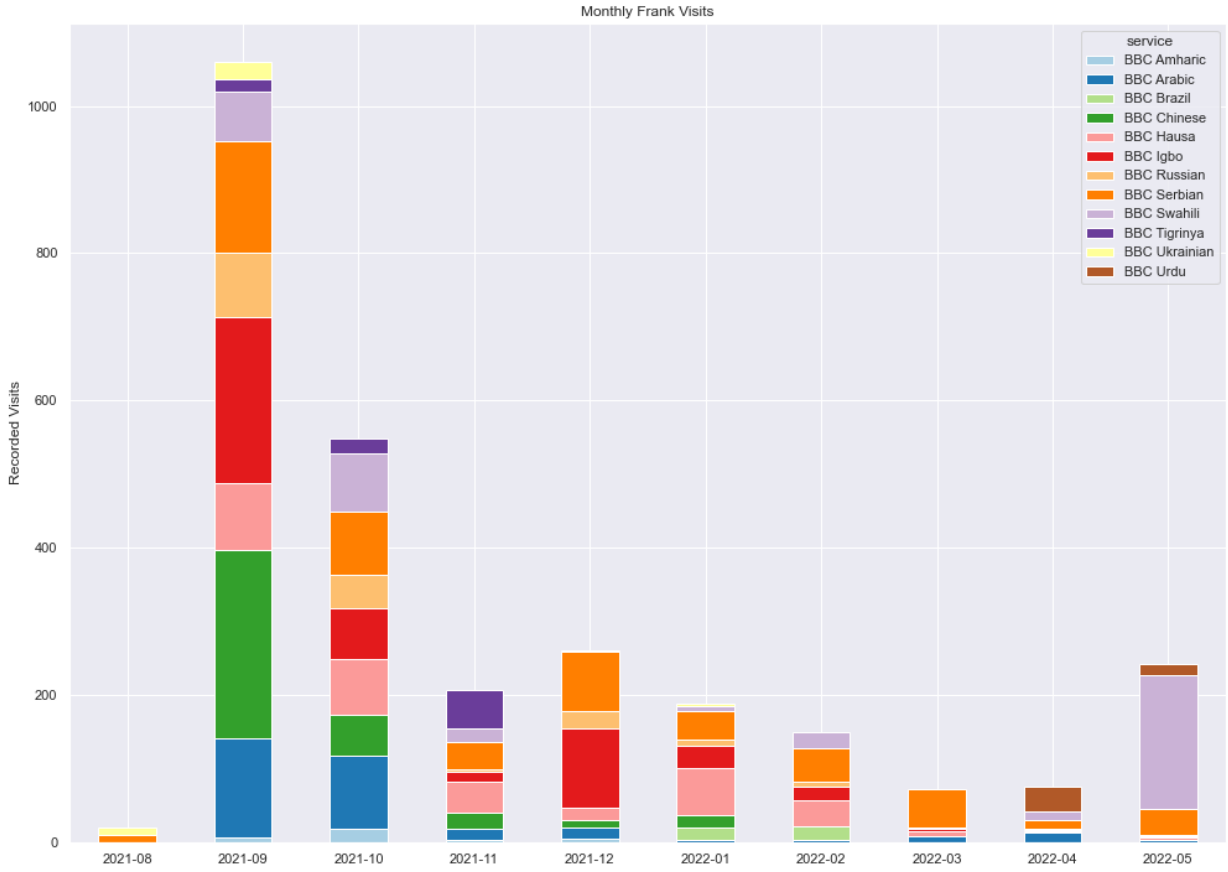

Figure 24: Frank usage by team Aug 2021 to May 2022. The spike in September 2021 corresponds with the formal trial period

What the trial has demonstrated is a clear need to have a visible means of assurance about the quality of content. This could be something as simple as a tick to show a journalist has checked and verified the translations provided, or alternatively an estimate of accuracy in percentages as provided by BBC's transcription tool or other external translation interfaces as noted in D5.2 (4.1.1 Functional requirements) or by providing a selection of alternative translations, which may be ordered by predicted likelihood. Nevertheless, future solutions have to be scoped in a way to mitigate any fear of reputational risks.

6.3 Multilingual Graphical Storytelling (Multilingual GST)

6.3.1 Background and use case (change of domain)

The GoURMET Grant Agreement included a third use case on International Business News Analysis. This involved 'reliably translating and analysing news in the highly specialised financial domain with a view to create a potentially commercial product'.

Due to a review of needs, priorities and opportunities, a joint decision was taken by the consortium, on the suggestion of the media partners, to revise the third use case to focus on improving translation models for the health domain instead of business.

There were a number of reasons for this:

- The original proposal did not fully take account of media user needs, specifically that there is less editorial need for business and a strong editorial need for health.
- Following the recruitment of a new set of project team members at the BBC in autumn 2020, there was a renewed focus on ensuring the goals of the project aligned closely with the needs of media content creators.
- As health is premium content with a high commercial value, so the costs of buying in health content are high, and subsequently the cost savings from generating more health content internally could be much greater than for business.

The BBC conducted an audit to sense check the usefulness of developing a domain model specialising in business for the BBC World Service.

The first step of this audit was identifying source material for potential translations. We identified that BBC News's digital content on business and economy is heavily skewed towards the needs of UK audiences, providing limited opportunities for translation for a global audience.

The second step was identifying potential outlets providing global business material intended for world audiences. These included World Business Report and World Business Update (business daily programmes on BBC World Service radio), Cash Éco (a business segment on BBC Afrique and BBC Pashto TV although ceased at the time of the audit due to Covid-19 measures), as well as business index pages and content volumes on BBC World Service websites.

As of 21 December 2020, out of approximately 40 language sites (excluding English, Welsh and Gaelic), only nine had a business index. Of these business indexes, three (Burmese, French, and Somali) produced less than two articles per week. Only three teams working on better-resourced languages (Persian, Portuguese and Spanish) had more than ten items per week in their index.

By contrast, 14 teams had a dedicated health index, and three had a science index including health content. Due to the Covid-19 pandemic at that time, all of the BBC World Service teams were producing unprecedented volumes of health-related stories in their daily coverage and on their digital front pages. Looking forward, the interest in health is set to continue with Covid-19 still endemic in communities, and new health issues arising (e.g. monkeypox).

The BBC's internal audience surveys also indicated that health was among the most underserved domains relative to audience demand.

Comparing commercial products, external providers such as Amazon and Google are offering improved translation capabilities for the health domain but these are priced at premium rates.

For all of the reasons above, investing in improving translation capabilities in the health domain offered a better return.

The GoURMET media partner DW agreed this would also be more helpful for their goals.

Therefore the decision was taken to switch the use case to health, with discussions on the details of the language or languages to be included to be deferred to a later date.

6.3.2 Languages

In September 2021, the initial thought was to select one high-resourced and one low-resource language for comparison. The data sets initially suggested included French, Spanish, Russian, and Chinese. As public media providers, the level of complexity in medical terms we would need to deal with would be limited.

Having run through some sample articles in these languages using generic models, it looked like the baseline generic models were already dealing well with the common medical terms. Therefore the potential benefit of investing further in these languages appeared to be limited.

As a result of these discussions, the academic partners suggested selecting Turkish as one language that would benefit from domain specialisation. It also presented an interesting research challenge due to the agglutinative nature of the language. Sampling existing articles confirmed there was significant room for improvement when translating health-related content.

It was proposed that the prototype should also include a limited number of other BBC World Service languages using baseline GoURMET and non-GoURMET models. There were four key considerations in shortlisting the languages to include those:

- Using the Latin alphabet to minimise potential additional dependencies relating to typographical differences and panel localisation – this ensured the workflow as a concept could be tested with minimal changes to the artwork as some of the panels have text on the picture (e.g. signs on buildings) which may not display correctly in other alphabetical systems
- With BBC teams with existing health content or an interest in health content
- With BBC teams likely to have staffing capacity to try out the prototype
- Supported or in scope to be covered by GoURMET (as Multilingual GST was primarly built with the aim of demonstrating these)

As a result we included five GoURMET languages (each language paired with English):

1. Swahili

6. Serbian

10. Hausa

11. Igbo

18. Turkish v2+ health domain adaptation (for more about this model see section 8.3).

We also included nine non-GoURMET languages (each language paired with English):

– Azeri

– French

– Gaelic

– Indonesian

– Portuguese

– Somali

– Spanish

– Vietnamese

– Welsh

6.3.3 Goals

Graphical Storytelling (GST) was an exploratory project first launched in 2019 to test the technical feasibility of generating semi-automated graphical news stories. It represents BBC News Labs' key goal of enriching and diversifying audience experiences via different formats.

Graphical news stories are multi-panel graphical depictions with text elements in the style of comics, manga or graphic novels that communicate a news story. They are particularly useful for a significant segment of BBC World Service audiences where mobile penetration is high but literacy is low.

Typically, illustrations are hand-drawn and take days or weeks to produce. This timescale doesn't align with newsroom requirements where multiple news articles are published in a day.

The commissioning process poses particularly acute challenges for smaller teams, who do not have in-house designers and are often deprioritised by graphics task planners when competing against requests from higher-impact outlets.

Multilingual GST is an extension of the original GST concept described above, using MT so stories in a variety of input languages can be translated into English as a pivot language to benefit from existing machine learning solutions. This translation can then be used to drive the text-to-image mapping process that creates graphics for stories to then be published in the original input language.

We saw GST as a perfect match for the health domain use case because health reports include a greater number of facts and figures that benefit from a more graphical treatment.

The goal therefore would be to make use of the existing GST infrastructure to allow journalists to publish a graphical story in one or more of the GoURMET languages.

GST is the only BBC prototype that is not bi-directional. It only translates content into English, which is used for the image generation process, with output text in the non-English language remaining intact in display.

6.3.4 Architecture info

The GST panel lifecycle (Figure 25) describes the processes and components involved in generating a graphical story:

1. A journalist writes a story (5-7 paragraphs) in the Story Editor.
2. If the story input is in a non-English language, the story is translated into English using Google Translate or GoURMET.

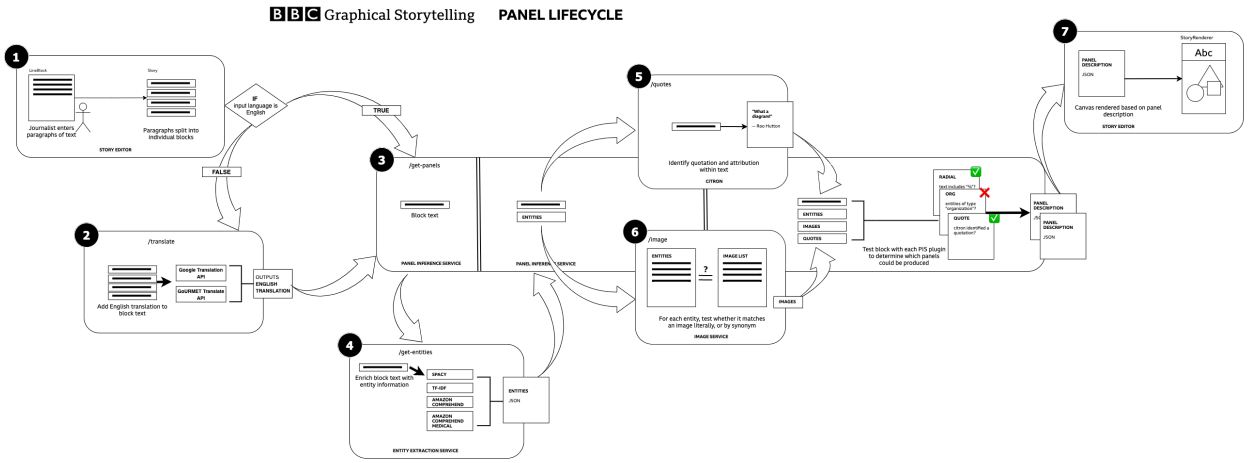

Figure 25: Graphical Storytelling (GST) panel lifecyle

3. The Panel Inference Service generates panel templates for each paragraph.
4. Key words and phrases (entities) are extracted from text.
5. Images associated with entities are pulled from image banks via an Image Service.
6. Quotes are extracted.
7. The Panel Editor screen in the Story Editor allows template layout and image modification.

The GST workflow is not fully automated since it requires the text to be optimally condensed for space by a journalist in the output language to fit in the limited space available with most impact.

The Story Editor has two views: a main view from which a journalist can start creating a story by adding narrative text, and a Panel Editor view where a journalist can choose between alternative graphics panels at the click of a button. Once the storyboard is complete, the panels are downloaded for publication.

6.3.5 Outputs

Figure 26 shows the Multilingual GST prototype user interface.

Below the graphics panels are two boxes. The box on the left controls the non-English text as it will appear on the panels. The translation is triggered automatically each time the text box content is changed.

The machine translation appears on the right. The user can correct or amend it to generate different image options without affecting the original text (in this case the original text being the input language for the translation and crucially also the final text that will appear on the non-English output).

Each panel has a small icon on the top left corner, which takes the user to the Editing View which offers variations of images derived from the sentence, as well as means to edit certain words (name of person, location) that are part of the image.

The Panel Renderer uses Paper.js to render panel templates returned from the Panel Inference Service to a canvas element. There are six panel template layout types:

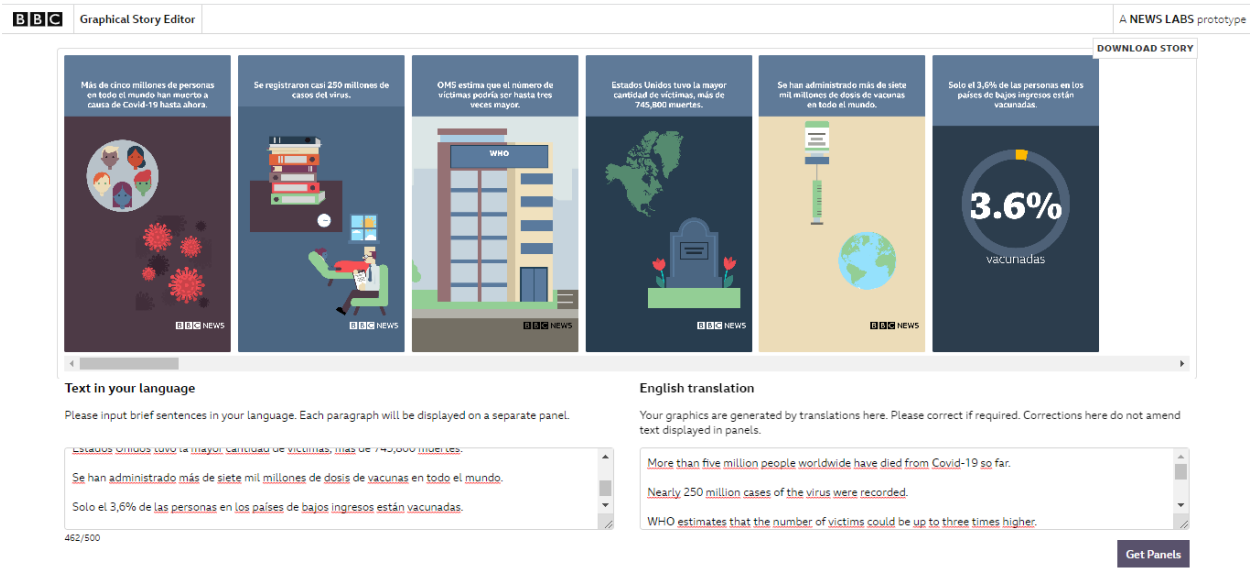

Figure 26: GST main view

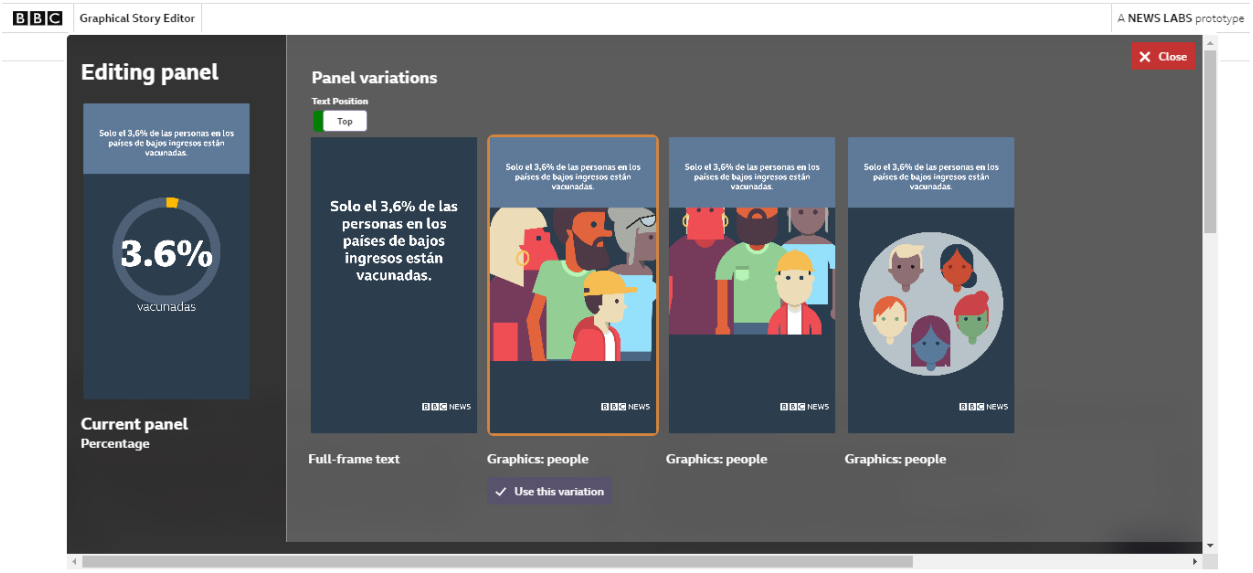

Figure 27: GST editing panel where basic words or image style can be amended

- Quote
- Org
- Radial
- Ratio
- Objects (graphics)
- Fallback (full frame text)

Objects panels are default for any paragraph where we can match entities to the images in the image bank. If quotes are detected we generate a **quote** panel (or an **org** panel if a business

is attributed to the quote). If a percentage is detected a **radial** panel will be generated and if a proportion is detected, a **ratio** panel is generated.

Once all necessary amendments are made, the user clicks the 'Download Story' button to render the graphics and download individual panels in a zipped folder containing .png files.

6.3.6 Findings from the project

The initial user trials were conducted when GST was first commissioned with the Indonesian Service who were keenly interested in health content. The prototype was also shared with the users from several BBC World Service teams in demo meetings and show and tells between December 2020 and December 2021. The informal feedback involved responses such as 'this is magical' and 'when can we start using it?'.

Despite an obvious consensus from the users about the feasibility of the idea, we have not been able to run a full, audience facing trial due to objections from the social media team who are gatekeepers to any content that appears on BBC's social media accounts.

The objections did not relate to the workflow, the usefulness of the tool or the quality of translations. The work came at a time when the social media team was overhauling the entire range of templates and designs they were using. They were highly averse to hosting any content that might diverge from the designs that were in the pipeline, and allowing teams to use them in an audience facing fashion even in a tightly controlled experiment.

At a time of unprecedented resource shortages due to the fallout of Covid-19, it was not feasible to run further trials with journalists unless the output of their efforts could be used as part of their offer.

Meanwhile, BBC News Labs managers were reluctant to agree to any changes to the GST design that was developed at extensive time and cost with internal and external UX input to alleviate the concerns.

Therefore, a decision was taken to continue with the work on Multilingual GST to ensure it is ready to share with interested stakeholders on demand for further development and potential productionising, but to cease the attempts towards a trial until the design rollouts could be completed and resourcing levels went back to pre-Covid levels.

The overall feedback we have received in user events indicates there is appetite to develop solutions that can help language teams generate and reversion graphics quickly and cost effectively. NLP and machine learning have a positive role to play in these processes.

Solutions such as Multilingual GST can:

- Help journalists generate content in image-poor domains such as health, business and sport
- Serve underserved audiences who prefer consuming visual rather than text content

Future iterations of GST can explore options to integrate the workflow with image search and curation tools so that the workflow can be replicated for a mix of images and graphics. It would also be interesting to experiment with a range of image transitions and exporting options.

A key learning from this process was ensuring involvement of stakeholders, particularly gatekeepers much earlier in the process to ensure a holistic understanding of what is deemed 'fit for purpose'.

Another conclusion, which applies to all three BBC prototypes, was the obvious need for efficient stakeholder and expectations management. BBC News Labs is not directly responsible for core workflow, investment or product decisions for the end users. Editorial stakeholders needed to be included as early in the requirements gathering process as possible, all the way down to the user trials with a clear road map.

Additionally, more communications were required about the differences between prototypes and products. A lot of the unfavourable opinions voiced, referred to the absence of certain features that were not deemed must haves for an MVP or prototype.

7 DW Prototypes

In this section we describe how the GoURMET models have been implemented and tested in prototypes that DW has built or selected as its human language technology (HLT) playground. Each prototype is either a dedicated GoURMET tool or has been further developed and enhanced during the GoURMET project to provide a test ground for GoURMET translation models and GoURMET project use cases.

7.1 plain X

7.1.1 Background and use case

The GoURMET engines have been implemented in the news.bridge platform, and its successor, plain X. news.bridge and plain X are enhanced AI toolboxes for multilingual audiovisual adaptation of content developed by DW and its Lisbon-based NLP development partner Priberam. The newer version, the plain X platform, is currently being rolled out in DW and beyond.

plain X is not a GoURMET-specific system, but a prototype that provides an infrastructure for testing and planning of use cases and exploitation. Further development, enhancements and initial roll-out occurred during the GoURMET project. It covers four main multilingual adaptation processes: enrichment of text, audio and video with semi-automated transcription, translation, subtitling and voice-over.

By means of a user-friendly UI, the editor (or other user dealing with multilingual content), uploads a video (or audio or text source) into the platform. (1) The video or audio content is converted to text using automated transcription. (2) A translation task can then be built upon that (source-language) text output, selecting a target language and an MT engine that has been incorporated into the platform. (3) The translated text can be formatted and displayed as properly segmented subtitles. (4) The translated text can also be read out by one or more synthetic voices in the selected target language.

The outcome can be exported from the platform as text files (for transcription or translation, for instance), as subtitle format (e.g. SRT or VTT) for closed captions or ingestion in post-production tools, or as embedded subtitles in the video (open captions).

As this is used in the actual production process of an international broadcaster, in particular DW, the initial automated process is enhanced with editorial control. Thus, the text is translated by the selected engine, for instance GoURMET English-Amharic, and then post-edited by a bilingual editor before it is finalised for publication. This means that ultimately the level of automation is at the discretion of the editorial team. Any shortcomings of the MT engine can thus be corrected manually by the editor. Of course, the goal is to arrive at the highest possible quality output by the different engines (ASR, MT and synthetic voice), requiring a minimum of post-editing effort.

Thus, by integrating the GoURMET engines into the plain X platform, with just a few clicks English-language video content can be subtitled and even voiced-over in any of the GoURMET languages (provided synthetic voices are available for that language) – with full editorial control to post-edit the MT output. Similarly, video content in any of the GoURMET languages can be subtitled and voiced-over in English.

7.1.2 Languages

Initially, in news.bridge, two languages were integrated and locally installed, using the dockerised engines. These two DW GoURMET languages were Bulgarian and Serbian, as they proved to be of reasonably good quality.

In plain X, all GoURMET languages in which DW produces content are being implemented, through an API accessing an AWS server, managed by DW, onto which the dockerised engines have been deployed by an NLP developer from the DW GoURMET team. This means: Swahili, Turkish, Bulgarian, Serbian, Tamil, Amharic, Hausa, Macedonian, Urdu, all into and from English. The other (non-DW) languages may be added later.

7.1.3 Goals

The plain X tool aims to enhance editorial multilingual workflows by offering an efficient one-stop-shop, thus avoiding the need to go into different tools for the different processes and aiming at offering the best quality available on the market.

This tool is expected to have a major impact on the editorial multilingual workflow of DW and other broadcasters/content providers who decide to onboard the (commercial version of the) platform, enabling them to combine and automate different steps, while leaving final editorial control with the editors. It will allow them to expand the language coverage and therefore widen their reach.

The goal is to cover a large variety of languages, including in particular low-resource languages, enabling, on the one hand, translation of content in such languages to other languages (especially also in English), making it available to a broader community – and thus supporting its wider dissemination and making its production more profitable. On the other hand, it also makes content from other languages (including English) available to smaller communities or those in remote areas, for instance, thus widening their horizon and opening up a world of content in their own language.

Another major goal is to contribute to an enhanced inclusion and accessibility. With plain X, content can be subtitled and voiced-over in virtually any language, including low-resource languages.

7.1.4 Architecture

It is an integrated platform for multilingual adaptation of video, audio and text content, combining different processes: transcription (ASR), translation (MT), subtitling and voice-over (synthetic voice).

For ASR and MT, the platform does not provide its own models, but serves as an integrated gateway to existing services (off-the shelf or customised). It uses third-party NLP engines for transcription, translation and synthetic voice. It offers a variety of engines and different aspects are taken into account for the selection, including quality of output, trustworthiness of the provider, reliability of access and stability, updating, cost, online and offline availability. Engines are only added if they have an added value, in the first place quality of course (i.e. if it is better than the other engines for one or more of the languages/language pairs).

However, there are other aspects to be considered: even if it ranks lower than other engines in terms of quality output, it can be added because it is less expensive, it is faster, it is available as an installable tool rather than API (or vice-versa), it is specialised in a certain domain.

Most engines are integrated using API access, but there are some locally installed engines, including initially GoURMET (now deployed and locally managed on an AWS server).

Defaults are/can be set for the selection of the engines. This selection is based on benchmarking (see below). This ensures that the most appropriate engine for that language (pair) is proposed to the user. Also within the text to be translated, the platform allows a smooth change to another engine per segment/sentence to compare the outcome.

Important for real productive work, the tool allows for collaborative work and review process, the latter being in particular important for translation. Working towards increasing accessibility is a major objective and this tool caters for easy and efficient subtitling in both source and (virtually any) target language. Customised templates allow for embedded branded subtitles in an in-house style. Of course, subtitles can be easily exported as closed captions, essential for highly multilingual content. The platform continues to be further enhanced, for instance to optimize language-specific and generic subtitle segmentation and to improve voice-over output.

plain X is integrated with internal data repository systems such as OpenMedia and with post-production systems. We have implemented different levels of automation and integration, including a live, fully-automated, source-language subtitling system. This collaboration ensures our requirements as an international broadcaster are at the centre of development and enhancements.

7.1.5 Outputs

Figure 28 shows the list of MT engines implemented in plain X for English into Bulgarian, featuring the GoURMET model.

Figure 29 shows the plain X interface. In this example, a video with a voice track in English has been subtitled in Serbian using a combination of Google for speech-to-text and GoURMET for subsequent translation.

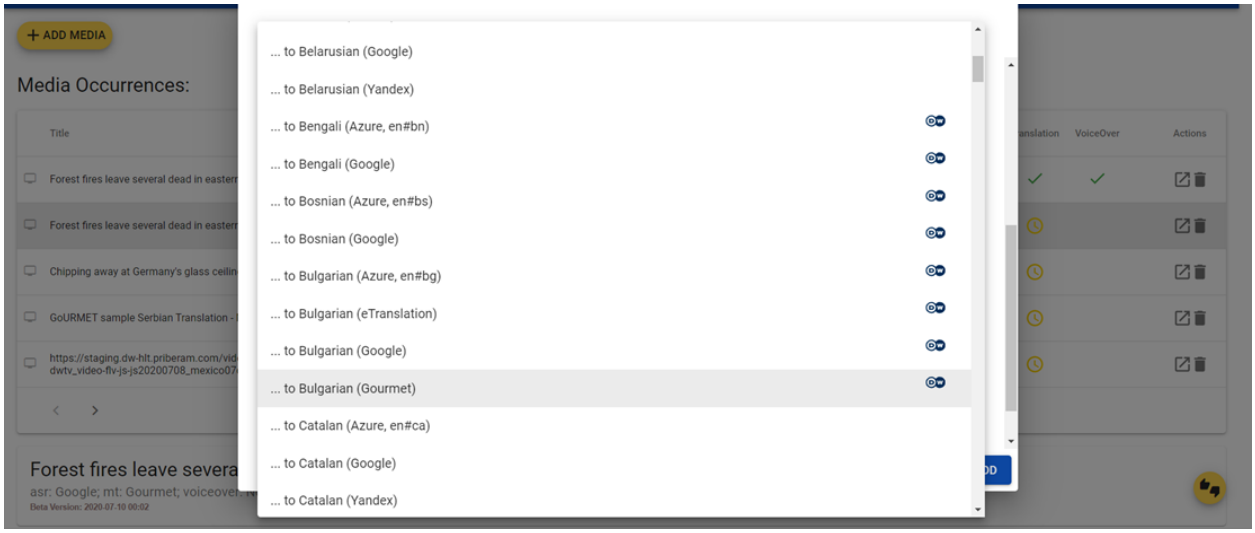

Figure 28: plain X interface showing GoURMET Bulgarian engines among the MT tools

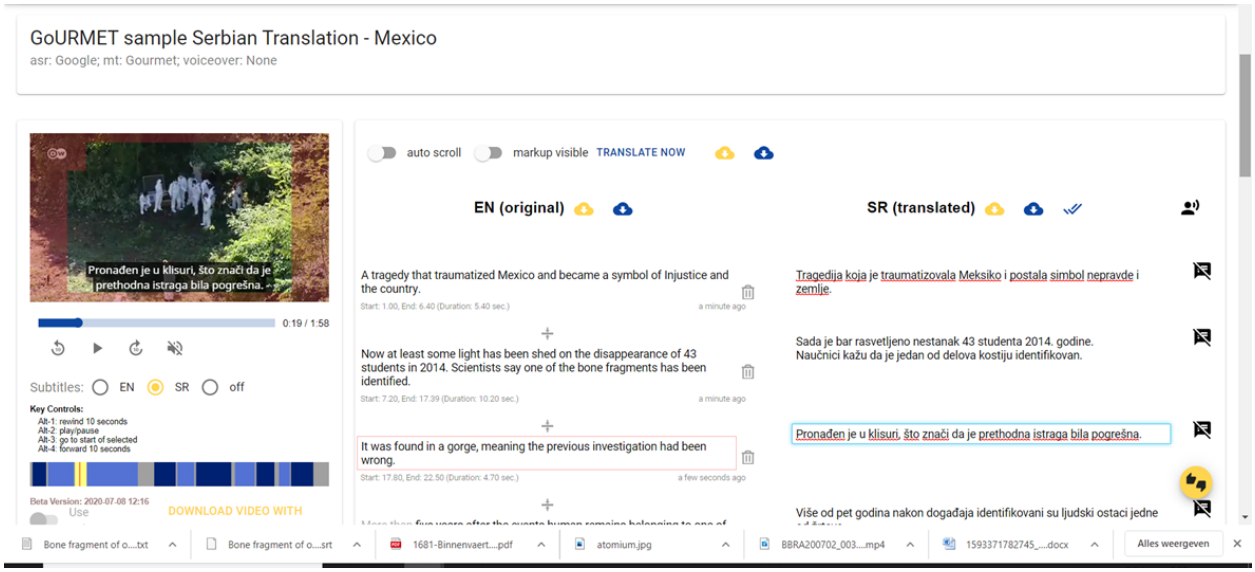

Figure 29: plain X interface showing a video translated and subtitled using GoURMET

7.1.6 Findings from user trials

7.1.6.1 Standardised integration

In order to facilitate adoption of new engines, it is important to allow an easy integration of such new (or updated) engines. Initially, the dockerised engines were not that easily integrated and integration was quite time consuming. A system was needed to speed up and standardise this process. A standardised process was set up for integration of the dockerised engines as well as the use of the engines via an API.

7.1.6.2 Updating and sustainability

To use MT engines in an operational, and in particular a productive, way, it is vital that they are updated on a regular basis to allow for integration of new terms. This was proven once more with the news coverage on the war in Ukraine, where frequently occurring new terminology, including place names, was mistranslated. Developing the models is one phase, which needs to be followed up with maintenance and updating. Otherwise the model soon becomes obsolete and not on par with other leading engines such as Google or MS Azure. This was raised in the consortium by the user partners and potential servicing by the technology partners was discussed.

7.1.6.3 Enhancements

Feedback from editors shows that initially many are reluctant to use machine translation for productive work, as they believe the quality is insufficient and there is also the fear that machines will eliminate the use of human translators/editors. However, once they have embraced such tools, they have high expectations and expect a near-perfect outcome. They seem surprised by the mistakes made by the MT. Thus, it is important to constantly enhance the quality, either by improving the MT engine itself, or by adding components to the MT output, such as named entity recognition and correction, integration of customised glossaries, synonym finders, and feedback.

7.2 SELMA platform

7.2.1 Background and use case

The GoURMET engines have been implemented in the SELMA protype. This protoype is being built in the framework of the SELMA EU-funded Horizon 2020 research and innovation project under grant agreement No 957017, focusing on multilingual media monitoring, in which DW, Priberam, the University of Latvia, the University of Avignon and Fraunhofer IAIS cooperate.

SELMA aims to develop a self-learning AI platform to analyse large volumes of data streams in over 30 languages and performs state-of-the-art research in NLP technologies and builds components to deploy such technologies. It applies MT at various stages, for instance to translate the source text into English before running NER and NEL components, if these are not available in the respective source language. Similarly, MT is applied to translate the textual outcome into English, the lingua franca, for analysis output and visualisation, reports and newsletters, upon request or based on user settings.

Some of the GoURMET engines are used and made available in several parts of the SELMA platform: the analysis tools, in the (proprietary) platform plain X (see further), and as part of the SELMA Open Source System (OSS) HLT UI, providing a simple tool for transcription, translation and voice-over of text, audio and video content. To enable free, open use of the open source SELMA use case, it only uses free HLT software and engines, thus the GoURMET engines are particularly suited for this purpose.

This means that text, audio and video can be processed and translated using the GoURMET models as they are implemented in the SELMA OSS. Transcribed or ingested texts can be translated and also rendered as audio using a synthetic voice.

7.2.2 Languages

In SELMA, all GoURMET languages were integrated into the platform. Table 12 indicates the different open source modules available on the SELMA OSS and the language coverage by GoURMET and other models.

7.2.3 Goals

SELMA is a broad HLT research and innovation project, coordinated by DW, that is aimed at advancing NLP technologies for the media industry in particular, including tools for comprehension and analysis (media monitoring) and content creation (publishing translations, generating subtitles or voice-over). Parts of it are developed as open source and other modules are proprietary, directed at specific business applications and markets. The overall integration architecture and basic NLP demonstrator are open source. This is where the use of open-source MT engines, such as GoURMET models, comes into play.

SELMA is also aimed at supporting the enhancement of low-resource languages through transfer learning and at evaluating the output for such languages and widening their reach. It builds further upon existing models and addresses the need for user feedback from post-editing to enhance the models. SELMA targets customisation through the creation and use of glossaries at different levels (user, team, organisation).

The prototype also works towards enhanced accessibility, by enabling and advancing the use of (source language and translated) subtitling (with speaker labelling and audio descriptions).

7.2.4 Architecture info

Figure 30 shows the SELMA architecture. We see the basic integrated infrastructure, using Maestro. It includes a range of NLP workers, with commercial services (e.g. AWS, Azure, IBM) as well as open source engines, of which GoURMET is one. Also some of the open-source models resulting from the SUMMA H2020 project are integrated.

7.2.5 Outputs

The dockerised engines of all GoURMET models have been implemented in the open-source module of the SELMA platform. They are part of the collection of NLP workers.

Figure 31 shows the SELMA OSS worker list. The lines starting with “type”:gourmet-mt show the different GoURMET engines that are available on SELMA.

Figure 32 shows the SELMA active workers. In this example we see that the English-Gujarati model is used to produce a translation.

Figure 33 shows an example of a translation from English into Turkish, executed using the SELMA OSS. The output can be adapted to user needs. In the example shown it is being translated be sentence by sentence, with the source language sentence followed by its translation into the target language. Alternatively, the tool can also be for an entire text (i.e. entire text in source language followed by the entire text in the target language). A user-friendly iPad app for such processing is also being developed.

Task	Transcription			Punctuation	Translation		TTS
Variant	Kaldi		Wav2Vec	Pause-based	M2M-100	GoURMET	VITS
Origin	LETA	SUMMA	Hugging-Face	LETA	Facebook	GoURMET	LIA
DW language							
Albanian				SQ	↔		
Amharic				AM	↔	↔	
Arabic				AR	↔		
Bengali				BN	↔		
Bosnian				BS	↔		
Bulgarian				BG	↔	↔	
Chinese Simple				ZH	↔		
Chinese Trad.				ZH	↔		
Croatian				HR	↔		
Dari				FA	↔		
English		EN	EN	EN	↔		
French				FR	↔		
German				DE	↔		
Greek				EL	↔		
Hausa				HA	↔	↔	
Hungarian				HU	↔		
Hindi				HI	↔		
Indonesian				ID	↔		
Kiswahili				SW	↔	↔	
Macedonian				MK	↔	↔	
Pashto				PS	↔	↔	
Persian				FA	↔		
Polish				PL	↔		
Portuguese BR				PT	↔		
Portuguese AF				PT	↔		PT
Romanian				RM	↔		
Russian		RU	RU	RU	↔		
Serbian				SR	↔	↔	
Spanish				ES	↔		
Tamil				TA	↔	↔	
Turkish				TR	↔	↔	
Ukrainian				UK	↔		
Urdu				UR	↔	↔	
Other languages							
Burmese				MY	↔	↔	
Gujarati				GU	↔	↔	
Igbo				IG	↔	↔	
Kirghiz				KY		↔	
Latvian	LV			LV	↔		
Tigrinya				TI		↔	

↔ indicates translation to and from English

Table 12: SELMA languages

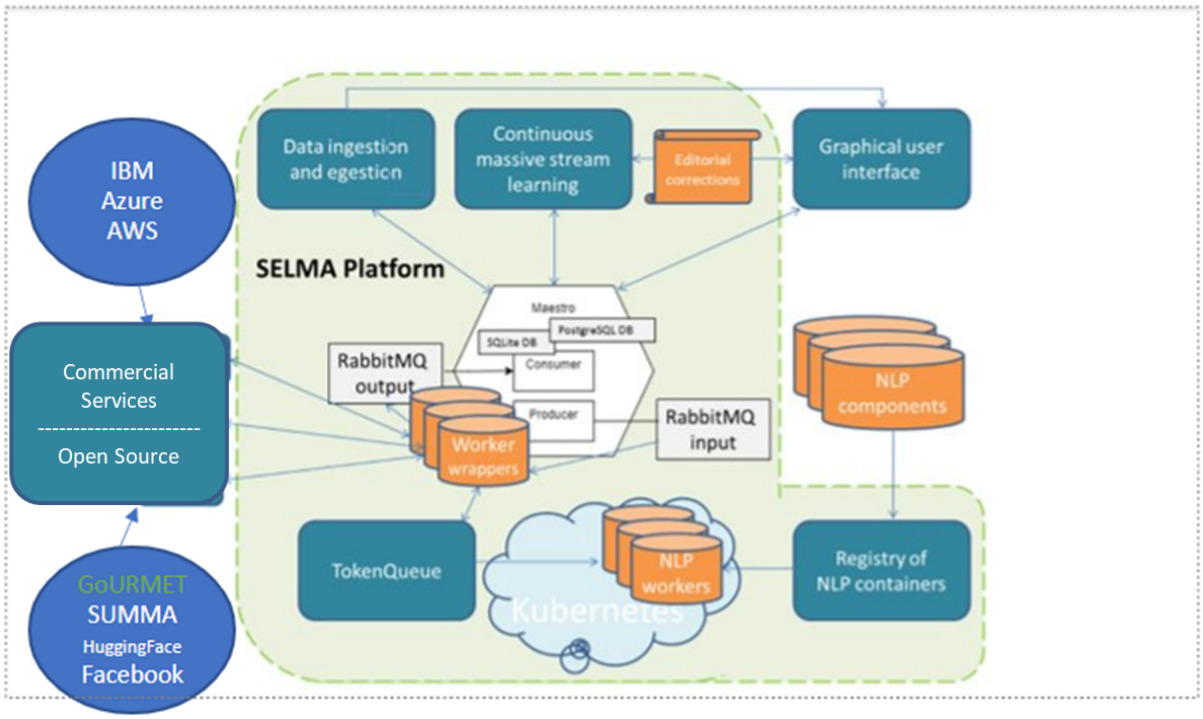

Figure 30: SELMA architecture

Worker List:

- { "type": "asr", "engine": "lv", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/asr-lv/post" }
- { "type": "asr", "engine": "en", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/asr-en/post" }
- { "type": "asr", "engine": "ru", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/asr-ru/post" }
- { "type": "w2v", "engine": "en", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/w2v-en/post" }
- { "type": "w2v", "engine": "ru", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/w2v-ru/post" }
- { "type": "segmenter", "engine": "default", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/segmenter/" }
- { "type": "mt", "engine": "lv-en", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/mt-lv-en/translate" }
- { "type": "tts", "engine": "br", "version": "0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/tts/api/tts" }
- { "type": "gourmet-mt", "engine": "en-bg", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-bg/translation" }
- { "type": "gourmet-mt", "engine": "bg-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-bg-en/translation" }
- { "type": "gourmet-mt", "engine": "en-gu", "version": "v0.2", "busy_since": 1655003975, "url": "http://194.8.1.235:8888/gourmet-mt-en-gu/translation" }
- { "type": "gourmet-mt", "engine": "en-ha", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ha/translation" }
- { "type": "gourmet-mt", "engine": "en-ig", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ig/translation" }
- { "type": "gourmet-mt", "engine": "en-ps", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ps/translation" }
- { "type": "gourmet-mt", "engine": "en-sr-cyr", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-sr-cyr/translation" }
- { "type": "gourmet-mt", "engine": "en-sr-lat", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-sr-lat/translation" }
- { "type": "gourmet-mt", "engine": "en-sw", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-sw/translation" }
- { "type": "gourmet-mt", "engine": "en-ta", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ta/translation" }
- { "type": "gourmet-mt", "engine": "en-ur", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ur/translation" }
- { "type": "gourmet-mt", "engine": "gu-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-gu-en/translation" }
- { "type": "gourmet-mt", "engine": "ha-en", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ha-en/translation" }
- { "type": "gourmet-mt", "engine": "ig-en", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ig-en/translation" }
- { "type": "gourmet-mt", "engine": "ps-en", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ps-en/translation" }
- { "type": "gourmet-mt", "engine": "sr-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-sr-en/translation" }
- { "type": "gourmet-mt", "engine": "sw-en", "version": "v0.3", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-sw-en/translation" }
- { "type": "gourmet-mt", "engine": "ta-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ta-en/translation" }
- { "type": "gourmet-mt", "engine": "ur-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ur-en/translation" }
- { "type": "gourmet-mt", "engine": "am-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-am-en/translation" }
- { "type": "gourmet-mt", "engine": "en-am", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-am/translation" }
- { "type": "gourmet-mt", "engine": "en-tr", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-tr/translation" }
- { "type": "gourmet-mt", "engine": "tr-en", "version": "v0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-tr-en/translation" }
- { "type": "gourmet-mt", "engine": "en-ky", "version": "0.0.2", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ky/translation" }
- { "type": "gourmet-mt", "engine": "en-mk", "version": "0.0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-mk/translation" }
- { "type": "gourmet-mt", "engine": "en-my", "version": "0.1.0.faster", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-my/translation" }
- { "type": "gourmet-mt", "engine": "ky-en", "version": "0.0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ky-en/translation" }
- { "type": "gourmet-mt", "engine": "mk-en", "version": "0.0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-mk-en/translation" }
- { "type": "gourmet-mt", "engine": "my-en", "version": "0.1.0.faster", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-my-en/translation" }
- { "type": "gourmet-mt", "engine": "ti-en", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-ti-en/translation" }
- { "type": "gourmet-mt", "engine": "en-ti", "version": "v0.1", "busy_since": 0, "url": "http://194.8.1.235:8888/gourmet-mt-en-ti/translation" }

Figure 31: SELMA OSS worker list

```
{
  "type": "gourmet-mt",
  "engine": "en-bg",
  "version": "v0.2",
  "busy_since": 0,
  "url": "http://194.8.1.235:8888/gourmet-mt-en-bg/translation"
},
{
  "type": "gourmet-mt",
  "engine": "bg-en",
  "version": "v0.1",
  "busy_since": 0,
  "url": "http://194.8.1.235:8888/gourmet-mt-bg-en/translation"
},
{
  "type": "gourmet-mt",
  "engine": "en-gu",
  "version": "v0.2",
  "busy_since": 1655003975,
  "url": "http://194.8.1.235:8888/gourmet-mt-en-gu/translation"
},
{
  "type": "gourmet-mt",
  "engine": "en-ha",
  "version": "v0.2",
  "busy_since": 0,
  "url": "http://194.8.1.235:8888/gourmet-mt-en-ha/translation"
},
```


Figure 32: SELMA active workers

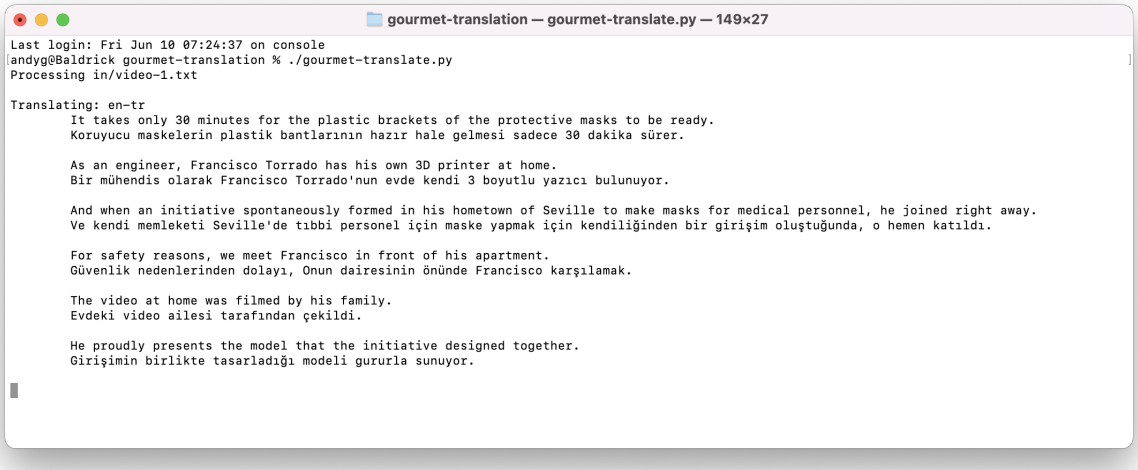

Figure 33: SELMA OSS backend processing

In addition, the GoURMET Translation UI, developed by BBC, has been integrated into the SELMA UI, giving direct access to the online GoURMET tool. This will remain in place as long as BBC continues to provide access.

7.2.6 Findings from user trials

7.2.6.1 Coverage

Since SELMA has many different use cases and prototypes, the engines, including GoURMET models, frequently run in the background and are not always visible to the user. Nevertheless, in order to give credit to the use of (in particular open source) models, this needs to become visible.

7.2.6.2 Implementation and integration

The integration of the dockerised engines was not particularly straightforward. An integration module was developed to facilitate engines to be deployed more easily. The same was done for API integration of HLT engines.

Four of the engines en-ky, ky-en, en-mk, mk-en reported an execution time of 0 seconds in their callback. Nevertheless, the translations are there, and no error is reported.

Some workers got stuck, likely linked to the SELMA Token Queue impementation (which will be replaced with a Reverse Proxy implentation to avoid these problems in the future). The process was quickly and easily restarted.

The system allows for sentence-by-sentence translation as well as entire text translation.

7.3 DW Benchmarking Tool

7.3.1 Background and use case

As DW is gradually enhancing its multilingual editorial workflows with AI, including ASR and MT, it is vital that we have a good overview of the quality output for the engines used for such processes. In particular since we apply these technologies and engines for all DW languages (currently 32), a thorough benchmarking effort, involving native speakers among the editorial staff for each language, as well as an efficient benchmarking system to support this, is required.

Thus, we have set up a sustainable internal benchmarking system, which compares the output of different engines and technologies, looks at various aspects of ASR, MT and synthetic voice. It is and will be further automated as much as possible, so that updated or new engines can efficiently be re-evaluated with a minimum of effort. Benchmarking is an ongoing process, otherwise the results soon become obsolete.

7.3.2 Languages

All 32 DW languages are covered by the benchmarking effort, including the following GoURMET languages: Swahili, Turkish, Bulgarian, Serbian, Tamil, Amharic, Hausa, Macedoniana, Pashto

and Urdu. In each case translation is to and from English. The other (non-DW) GoURMET languages will also become part of the benchmarking later.

7.3.3 Goals

The goal is to ensure we guide the editors in the best possible way in applying the most appropriate AI engine for the intended purpose.

Our aim is primarily qualitative assessment, in addition to some automated processes using WER and BLEU. We use one primary text that is translated in all (DW and some other) languages, so we get a consistent, comparative result and can use it for all possible language combinations. For some major languages, we extend this to five documents, in order to cover different types of text (general news, scientific, interview, etc.) and compare the output.

Consequently, one model translation is created from English or German into a target language, for instance Urdu. The same is done (using the same text) for other target languages, e.g. Arabic and Macedonian. Consequently, we can use the same texts for all possible language combinations covered, for example Arabic-Urdu or Macedonian-Arabic. This method allows us to cover an extremely high number of language combinations using one model text.

The evaluation covers the three main NLP technologies used, i.e. automated transcription (ASR), machine translation (MT), and synthetic voice. A basic automated, technical evaluation is done for MT and ASR. Human evaluation is done for all three processes. More details are provided in the section below.

7.3.4 Architecture info

A combination of automated and human evaluation is used. Since the different processes (transcription, translation, voice-over) are interrelated in the DW applications, as for instance the quality of the ASR output also affects the MT output in a video translation, we address the different steps in the benchmarking process here, not only MT. Figure 34 shows the Benchmarking architecture.

7.3.4.1 Naming convention for files

The following format is used when saving files:

"[video name]_[task]_[sourcelanguage]_[targetlanguage]_[engine]_[region].txt"

Example:

"Video1_MT_DE_EN_Google_UK.txt"

7.3.4.2 Transcription

The ranking is based on (1) an automated evaluation using word error rate (EBU WER rating tool) and (2) human evaluation.

WER (word error rate) is a widely accepted method to measure the quality of automated speech recognition (ASR). Basically, WER is the number of errors divided by the total words. It needs

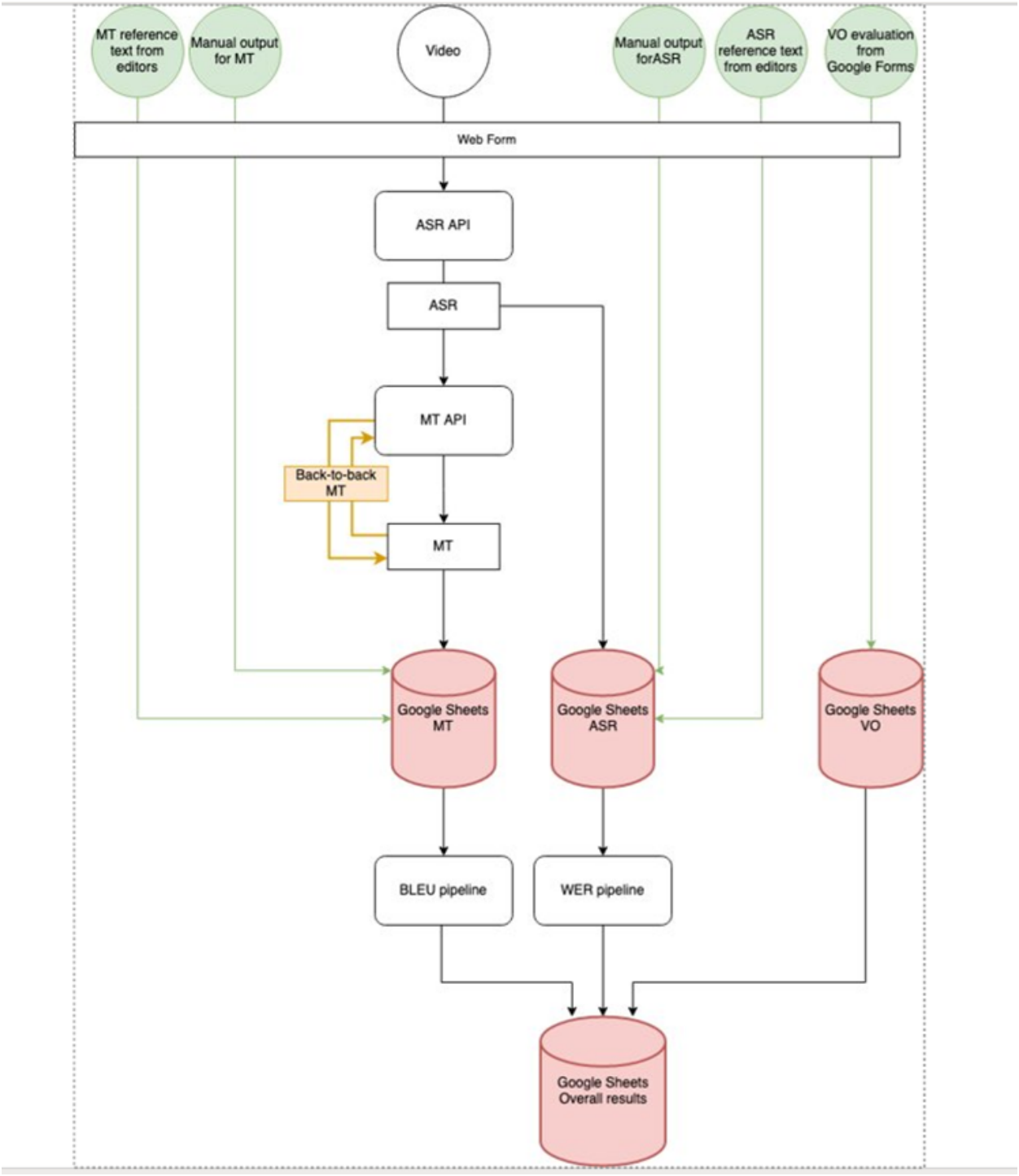

Figure 34: Benchmarking architecture

a reference text (a perfect transcription) as a model. The tool analyses the ASR output text, adds up the substitutions, insertions, and deletions that occur in the text and divides that number by the total number of words recognised from the spoken words in the audio.

For WER: the lower the score, the better (measured in percentage).

For the human rating: the higher the score, the better (max score = 5). The following aspects are being rated in terms of ASR:

- Language variety
- Accuracy
- Punctuation
- Capitalisation
- Completeness
- Background noises
- Speaker gender identification
- Speaker diarisation
- Timecode accuracy

7.3.4.3 Translation

The ranking is based on (1) an automated evaluation using BLEU score (Tilde BLEU rating tool) and (2) a human evaluation. The higher the score, the better.

BLEU measures how many words overlap with a reference translation, giving higher marks to sequential words. BLEU scores range from 1 to 100%. Less than 15% is not good. Around 50% is a good score. The goal is to get as close to 100% as possible – the closer to 100%, the closer it is to the reference. Thus, this automated method needs a parallel text (in two or more languages) that can be used as a reference translation.

Thus, for the automated benchmarking of MT, we use the BLEU score. It can easily be calculated using the interactive BLEU score evaluator from Tilde[1].

The human evaluation score is calculated based on user input from a Google Questionnaire. It takes the following aspects into consideration and users are asked to rate the aspect on a 5-point Likert scale:

- Accuracy
- Punctuation
- Capitalisation
- Fluency
- Completeness
- Language variety (i.e. does the translation reflect regional language differences)

7.3.4.4 Voice-over

For the benchmarking of the voice-over, we ask for user input by means of a Google questionnaire. At this point in time, as far as we are aware, there are no automated tools for assessing the quality of synthetic voices.

[1] https://www.letsmt.eu/Bleu.aspx

Thus, the ranking is based on human evaluation in terms of:

- Pronunciation
- Naturalness
- Rhythm
- Flow
- Melody
- Intonation
- Pitch

The higher the score, the better.

7.3.4.5 Automation

As explained earlier, the benchmarking process covers three tasks: Automatic Speech Recognition (ASR), Machine Translation (MT) and Voice-Over (VO). We assess ASR and MT using both algorithms and human input, whereas VO is only assessed by human evaluation.

For ASR and MT, we calculate the Word Error Rate (WER) and BLEU score, respectively and ask users to rate the model outputs. These two results are then averaged and combined in an overall score.

As manual computation is tedious and time-consuming, we aim to automate most parts of this benchmarking process. The WER and BLEU metrics are computed and stored in a spreadsheet automatically. The user only inputs the human evaluation for ASR, MT and VO.

We are enhancing this system to allow the user to upload content (a video file, audio file, or text) to be transcribed and translated automatically, as required, without any other user input. This will call the transcription and/or translation APIs of the major commercial engines (e.g. Azure, Google) and other engines we wish to evaluate, such as GoURMET, which will provide us with an automated way of benchmarking languages for which human evaluation is difficult.

This same pipeline will also help benchmark MT for low-resource languages by performing back translation which provides a basis for technical evaluation and an understandable output for human evaluation in the source language.

For this, we translate our model text from English to a predefined target language and we translate the output back into English, using MT. This output can then be evaluated using both the BLEU score and a human assessment. This is particularly useful to evaluate the quality and therefore usefulness for languages not currently covered in the organisation, so for which DW does not have native editors, or to do a quick and automated benchmark for which a reference text is not (yet) available in the target language or editors are not available for assessment.

Although this technique assumes that the source-to-target translation is somewhat equivalent in quality to the target-to-source translation, which is not necessarily the case, it provides a satisfactory result for both human and automated evaluation of low-resource languages.

7.3.5 Findings from user trials

7.3.5.1 Limited scope

Since we are aiming to cover all possible language pairs between a large number of languages (32 DW languages and more), we use 1 to 5 standard texts translated into all target languages and reuse that for any of the language combinations. This pragmatic and consistent way of working of course gives a limited view of the output quality, as it only considers those few texts.

We realise this does not in any way meet the expectations of quantitative evaluation, but that is not the goal here. That is covered by the technical partners. The consistency in our benchmarking system gives us the possibility to compare the output among the different targeted languages.

7.3.5.2 Model

The benchmarking requires creating a model, a reference to be used as a baseline for each of the technologies and target languages. This needs commitment from each of the language teams to translate the standard model text from the source language and produce the translation into the target language. For ASR assessment, a video is selected per source language.The challenge here is to ensure the type of content and difficulty level is similar for all languages covered.

7.3.5.3 Variable outcome for MT

It should be taken into consideration that the reliability of measurable output varies with the technology assessed. For ASR, the outcome is pretty straightforward: the ASR output should match the model almost to perfection. It is either right or wrong. Exceptions could include different (accepted) spellings of named entities, punctuation, including hyphenation, and abbreviations.

For MT, it is a completely different situation, as one text can have many different – correct – translated versions, also depending on the organisational or personal style. Therefore, a translation will rarely fully match the model text, but can still be a valid and correct translation. Thus, automated comparison using BLEU scores are only partly reliable. User evaluation overcomes some of these issues, but here personal preferences come into play, making the scoring quite subjective.

7.4 DW Local HLT Research Modules

The DW Research and Cooperation team has also installed the GoURMET engines in its infrastructure for further internal use, research and evaluation. As these models are locally installable, the flexibility and absence of running costs make them particularly suitable for HLT research and development.

They will be used for instance to run summarisation, named entity recognition, combined HLT processes, automated journalism, verification processes, user feedback integration, to name a few. Also the fact that this tool is locally installed makes it suitable for content that should not leave the premises for reasons of sensitivity or licenses.

8 Collaborative Improvement

This section offers three case studies of collaborative improvement where the research partners and the media partners worked on iterative changes to the models and the prototypes. They are some of the best examples of where the collaboration on GoURMET worked to solve problems and advance new solutions.

Two of the languages took longer to integrate than the others. This section looks at the work to integrate these two languages in particular. We have chosen to describe these two here as they highlight the types of issues we encountered and overcame.

8.1 Swahili↔English

Swahili was going to be the first language integrated with the prototypes, but due to significant issues ended up being the last. As such, it is a useful case study of patience, perseverance and collaboration.

The BBC was keen to deploy Swahili in live prototypes as it was one of the best performing languages in standalone tests, but when we came to do so it was far too slow for practical use. It took over a year to address the following:

8.1.1 Speed

The GoURMET deliverable D5.2 Use Cases and Requirements section 4.1.2 states that 'MT technologies shall translate a sentence in not more than 500ms'. Assuming an average sentence of ten words and an average news article of 500 words, a translation could still take up to 25 seconds to be processed to be deemed acceptable.

However, initial tests conducted on the models demonstrated they were often taking longer for one sentence (e.g. two seconds, as compared to the required 500ms or half a second). Viewed against a backdrop of improvements to Google Translate, which now takes less than a second to return a 500-word article, there was clearly room for improvement with GoURMET before a clear case could be made to exploit it in a competitive global newsroom setting, where speed and accuracy are paramount.

The BBC initially ran the model on AWS Fargate without GPU and in this context it was not fast enough to be used in the LPT prototype. The BBC raised this at a consortium meeting where various solutions were considered. There were three suggestions for further consideration:

- Pre-compute the translations – this would require adding a data store and for everything to be pre-translated instead of translated on demand
- Increase the number of ECS tasks – this would increase fixed running costs
- Auto-scale ECS tasks – but models take some time to load so this may not improve the responsiveness of the prototype

The outcome of these discussion led to an experiment in running the model on AWS EC2 with the instance type p2.xlarge (see aws.amazon.com/ec2/instance-types) with two groups of tests on EC2, one with GPU enabled and one without.

Further dialogue between the BBC and Alicante led to a number of performance improvements being implemented which included:

- Using a smaller beam, without sacrificing quality (a beam of 5 usually worked well)
- Using lexical shortlists, to reduce the time for the most expensive step of translation (softmax computation).
- Compiling the truecase model in advance to binary format (the initial version include a text model, which was compiled every time you start the docker, which takes time).
- Improving the threading/batching to better suit the BBC use-case (small documents), and providing command-line arguments to tune these parameters.

In later releases of Marian-based models, we applied quantisation to give further speedups, and replaced the preprocessing pipeline with a single-step SentencePiece pipeline. The lesson learnt here is that the research partners need to test the speed of translation before release, and this was incorporated into the process for future releases.

With speed improved, the BBC then tested the updated model further in order to integrate it into the Live Pages Translation (LPT) prototype. A few further issues were discovered that required support:

8.1.2 Multiple paragraphs

There was an issue with the requirement for translations to be split into multiple paragraphs. To clarify the use case, we wanted to be able to send a complete article with multiple paragraphs and the output to be structured in equivalent translated paragraphs. For example:

Input

> *Kenya imeripoti ongezeko la viwango vya maambukizi ya virusi vya Corona hadi asilimia 16 baada ya watu 731 kupatikana na ugonjwa huo siku ya Jumatatu.*
>
> *Takwimu kutoka Wizara ya Afya nchini humo, zinaonyesha kwamba watu 109 walikuwa katika kitengo cha wagonjwa mahututi huku 23 wakisaidiwa kupumua kutumia mashine.*
>
> *Kupitia taarifa Waziri wa Afya Mutahi Kagwe, amesema watu 4,513 ndio walipimwa virusi hivyo katika saa 24 zilizopita. Idadi ya waliombukizwa virusi hivyo nchini Kenya sasa ni 113,967 kutoka sampuli 1,373,839 zilizopimwa hadi sasa.*

Output expected

> *Kenya reported an increase in rates of transmission of the Corona virus to 16% after 731 people were found on Monday.*
>
> *Statistics from the Ministry of Health in the country show that 109 people were in intensive care with 23 support for machine breathing.*
>
> *In a statement, Health Minister Mutahi Kagwe said 4,513 people were tested in the last 24 hours. The number of people with the virus in Kenya is now 113,967 from 1,373,839 samples tested so far.*

The BBC considered two possibilities to achieve this:

- Send in one string using `\n` to separate paragraphs, but this was tried and the model seemed to ignore `\n`
- Send as an array of strings, one for each paragraph, which is the way that Google Translate works

The lead researcher on this model (Juan Antonio Pérez-Ortiz from the University of Alicante) advised that the current API indicates that there is one parameter `"q"` with the text to translate. The server splits the text into sentences, translates all of them in a single batch (if possible), and concatenates the translations for the response. Extra end-of-lines are therefore ignored.

As updating the API would require having to update all the released models, it was suggested that the best choice would be for all this preprocessing to be done on the client side. The plan was that the API should be as agnostic as possible (Unix philosophy) in the sense that all format-related operations are supposed to be done as external pre- or post-processing steps. In this case, one possible solution would be to have the translation segmented into paragraphs and to send each paragraph in a different request.

The BBC responded that this fix created further performance and speed issues. As a test, the following paragraphs were sent as separate requests:

> ***Paragraph 1:*** *Kifo cha Magufuli; Vijana walipewa fursa gani katika ulingo wa siasa za Tanzania?*
>
> ***Paragraph 2:*** *Wakati wananchi wa Tanzania wakiwa bado katika siku za maombolezo ya aliyekuwa rais wa nchi hiyo Dkt. John Pombe Magufuli, katika upande wa pili amewaandalia vijana ambao wanatarajiwa kuwa hazina ya uongozi kwa miaka ijayo.*
>
> ***Paragraph 3:*** *Magufuli katika kipindi chake cha miaka mitano alitoa nafasi kwa viongozi mbalimbali vijana katika ulingo wa siasa za Tanzania.*
>
> ***Paragraph 4:*** *Kuanzia mawaziri, wakuu wa mikoa, wakuu wa wilaya, makatibu tawala hadi kuwapitisha baadhi yao kuwania nafasi za ubunge katika bunge la Jamhuri.*
>
> ***Paragraph 5:*** *Kupitia nafasi ya urais hadi uenyekiti wa CCM, Magufuli ametengeneza vijana ambao wapo tayari kutumikia taifa hilo bila kujali rangi, kabila, dini na ukanda.*

On average, it took the model 939ms to translate paragraph 1, 1486ms for paragraph 2, 1344ms for paragraph 3, 1777ms for paragraph 4 and 2308ms for paragraph 5. This gives a total of 7854ms to translate all five paragraphs.

We then tested sending all the text as one query:

```
curl -X POST -d '{"q": "Kifo cha Magufuli; Vijana walipewa fursa gani katika
    ulingo wa siasa za Tanzania? Wakati wananchi wa Tanzania wakiwa bado
    katika siku za maombolezo ya aliyekuwa rais wa nchi hiyo Dkt. John Pombe
    Magufuli, katika upande wa pili amewaandalia vijana ambao wanatarajiwa
    kuwa hazina ya uongozi kwa miaka ijayo. Magufuli katika kipindi chake cha
    miaka mitano alitoa nafasi kwa viongozi mbalimbali vijana katika ulingo wa
     siasa za Tanzania. Kuanzia mawaziri, wakuu wa mikoa, wakuu wa wilaya,
    makatibu tawala hadi kuwapitisha baadhi yao kuwania nafasi za ubunge
    katika bunge la Jamhuri. Kupitia nafasi ya urais hadi uenyekiti wa CCM,
    Magufuli ametengeneza vijana ambao wapo tayari kutumikia taifa hilo  bila
    kujali rangi, kabila, dini na ukanda."}' -H "Content- Type: application/
    json" localhost:4000/translation
```

The average response time for this was 3233ms, which is much shorter than the total time of 7854ms above.

The assumption was that if we could send all the paragraphs as a list of texts and receive a list of translations, we could get the performance and speed benefits. The speed was crucial in order to integrate the model with a prototype focused on live pages.

It was suggested this may require a change to the API so that `"q"` could be an array, instead of just a string. This would also require a consequential change of the dockers. In the end the API didn't need to be changed as it already supported both strings and arrays, simply passing whatever it receives to the model. Instead the models required a small modification to accept arrays.

As a result of changes to the models to allow multiple paragraphs in a single request, response time was reduced from 8s to 3s on a typical article with 5 paragraphs.

8.1.3 Caching

The third issue that needed to be addressed related to the model providing incurred responses as a result of caching issues.

Testing the Swahili translation model using cURL, we found it would get into a state where it would return the output from the previous request, not the current request. It seemed like the model was returning a cached response.

Repeated tests would give the same request four or five times before the output would change. Restarting the model seemed to fix this, although it was not clear exactly what caused it.

It was suggested this could be something in the proxy configuration as Flask itself (used in the translation server) does not provide caching.

Run locally on a Linux box (hundreds of requests for translation of sentences containing a different number each) everytime gave a brand-new translation with the corresponding number in the input sentence:

```
for i in $(seq 1000 2000);
do curl -s -X POST -d "{\"q\":\"Mtu huyu atalipia kila kitu $i.\"}" -H "
   Content-Type: application/json" -v localhost:4000/translation 2>/dev/null;
done
```

The caching issue turned out to be due to a minor bug in the model, eventually identified by a member of the BBC team and was then swiftly resolved by the researchers involved who re-dockerised the models to prevent the behaviour in Swahili as well as Burmese, Pashto, Macedonian where the same issue was detected.

This is further confirmation of the well-known maxim that having a fresh perspective on a problem can often result in a breakthrough being made more quickly.

8.1.4 Hallucinations

There was a further problem with a behaviour known as 'hallucinations', which is a well-known problem of neural machine translation (NMT) systems (Raunak et al., 2021).

Hallucinations should not happen very often but if it really affects applications or disturbs human translators, it can be fixed with a heuristic such as removing all the words in the translation that get repeated more than x times (for example $x = 1.5$). We suspect that commercial systems perform this kind of cleaning in order to limit occasional repetitions.

Usually sentences containing hallucinations are not good translations even when repetitions are removed, but at least by removing the errors it makes what may not be an ideal result more manageable.

The origin of hallucinations in NMT is not completely known but the current predominant hypothesis is that things such as the amount or type of noise in the training corpus, the use of the system in a domain different than that of the training set, or the translation of sentences that radically do not follow common linguistic structures observed in the training corpus might be behind this behaviour. This may explain that hallucinations happen less or more in other models or situations. Sadly, there is no sound proposal on how to alleviate them.

8.1.5 Empty string error

Finally, we discovered that when the request string is empty ({"q": ""}), the model returns HTML instead of JSON (see below).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>500 Internal Server Error</title>
<h1>Internal Server Error</h1>
<p>blank lines are not allowed</p>
```

Ideally, we needed the model to be consistent and always return JSON, so we requested that the model return the following when the request string is empty:

```
{
        "error": null,
        "result": "",
        "time_taken": 9391.46
}
```

Or alternatively, if simpler to fix:

```
{
        "error": "blank lines are not allowed"
}
```

8.1.6 End result

As a result of the work above, version 0.4.0 of the English-Swahili docker images accepts both a single string or an array of strings in the request and besides this returns an empty translation for an empty input string. These are run with:

```
docker run --env BEAM_SIZE=5 --env BATCH_SIZE=32 --rm --ipc=host -p 4000:4000
   translation-sw-en:0.4.0

docker run --env BEAM_SIZE=5 --env BATCH_SIZE=32 --rm --ipc=host -p 4000:4000
   translation-en-sw:0.4.0
```

As an example, for these requests:

```
curl -X POST -d '{"q":["This person will pay for everything 1. This person
   will pay for everything 2. This person will pay for everything 3. This
   person will pay for everything 4.","This person will pay for everything
   5.","This person will pay for everything 6.","","This person will pay for
   everything 7. This person will pay for everything 8. This person will pay
   for everything 9.",""]}' -H "Content-Type: application/json" -v localhost
   :4000/translation

curl -X POST -d '{"q":["This person will pay for everything 90."]}' -H "
   Content-Type: application/json" -v localhost:4000/translation

curl -X POST -d '{"q":"This person will pay for everything 91."}' -H "Content-
   Type: application/json" -v localhost:4000/translation

curl -X POST -d '{"q":["","",""]}' -H "Content-Type: application/json" -v
   localhost:4000/translation

curl -X POST -d '{"q":""}' -H "Content-Type: application/json" -v localhost
   :4000/translation
```

The system would then return:

```
{"error":null,"result":["Mtu huyu atalipa kwa kila kitu 1. Mtu huyu atalipa
   kwa kila kitu 2. Mtu huyu atalipia kila kitu 3. Mtu huyu atalipia kila
   kitu 4.","Mtu huyu atalipia kila kitu 5.","Mtu huyu atalipia kila kitu
```

```
   6.","","Mtu huyu atalipia kila kitu 7. Mtu huyu atalipia kila kitu 8. Mtu
   huyu atalipia kila kitu 9.",""],"time_taken":1086.38}

{"error":null,"result":["Mtu huyu atalipia kila kitu 90."],"time_taken
   ":299.46}

{"error":null,"result":"Mtu huyu atalipia kila kitu 91.","time_taken":307.04}

{"error":null,"result":["","",""],"time_taken":0.05}

{"error":null,"result":"","time_taken":0.04}
```


8.2 Turkish↔English version 2

We revisited Turkish for a second time towards the end of the project as it is a language of joint interest for the BBC and DW. This is because it features very prominently in the news, and is one of the most spoken foreign languages in German households due to the large immigrant population.

At the time the language was selected we looked on the BBC News English website at the incidence of articles relating to the geographical regions for all languages under consideration for the cycle, and Turkish was one of the top languages featured.

The researchers were also interested in Turkish due to the agglutinative nature of the language posing a particular challenge, and they had learned a great deal since the first model was developed they were keen to revisit their work to apply their findings.

There was also an additional opportunity to try out different options swiftly since the BBC's GoURMET coordinator is a qualified Turkish translator and journalist and could help with testing without having to go through formal recruitment processes required for journalists.

The quality of the translations during the first pass were quite poor so we agreed as a consortium to redo the model based on what we had learned since the start of the project.

In a random sampling from December 2020, looking into one of the most read stories at the time, the approval for the first Covid-19 vaccine, the translation had several basic errors including translating 'Ingiltere' (which could have been translated as Britain, England, United Kingdom) as United States, as well as repetitions and issues with syntax.

Turkish input

> *İngiltere'de ilaç ve tedavilere onay veren denetleyici kuruluş İlaç ve Sağlık Bakımı Ürünleri Düzenleme Kurumu (MHRA) Pfizer/BioNTech aşısının acil kullanımına onay verdiğini duyurdu.*

GoURMET Turkish > English v1 output

> *The controller agency for the pharmaceutical and pharmaceutical care system in the UK, announced in the United States, has approved the immediate use of the Pfizer / BioNTh vaccine for the Office of Health Insurance (MHRA).*

Expected output

> *Britain's regulatory body for medicines and treatments, the Medicines and Healthcare Products Regulatory Agency (MHRA) announced it has approved the emergency use of Pfizer/BioNTech vaccine.*

The University of Amsterdam utilised MBart25 in the second round of work, which led to a significant improvement in the output quality as demonstrated in comparative sampling.

Turkish was then deployed in Frank (but not in LPT due to the size of the model slowing down return times) and was picked for further development in the Health Domain work explained below in section 8.3.

8.3 Turkish↔English version 2+ health domain adaptation

Despite the significant progress achieved with the new Turkish model, the output still had room for improvement, particularly in the field of health, with Covid-19 and related health issues dominating the coverage in news throughout 2020 and 2021.

Table 13 gives some sample translations demonstrating how GoURMET dealt with health-related terms prior to the adaptation work.

Table 13: Sample translations with health-related terms

Original	**Expected**	**GoURMET v2 (without terminology)**
They include MERS, Ebola, Dengue fever, plague, bubonic plague, Hantavirus, Zika, Rift Valley Fever.	MERS (Orta Doğu Solunum Sendromu), Ebola, Dang humması, veba, hıyarcıklı veba, Hantavirüs, Zika, Rift Vadisi ateşi bunlar arasında yer almaktadır.	Bunlar arasında MERS, Ebola, Diş çürüğü, çölyak, bubonik çölyak, Hantavirüs, Zika, Rift Valley Ateşi yer almaktadır.
Miscarriage may lead to a range of mental issues in women.	Düşük yapmak, kadınlarda bir dizi ruhsal soruna yol açabilir.	Hamilelik, kadınlarda zihinsel sorunlara yol açar.
The doctors decided to extubate the patient.	Doktorlar hastayı solunum cihazından ayırmaya karar verdiler.	Doktorlar hastayı buharlaştırmaya karar verdiler.

Samples provided in Table 13 provided hints of areas to focus on for developing a terminology list. For example, in line one, 'Dengue fever' was translated into Turkish as 'Diş çürüğü' (tooth decay) and 'plague' as 'çölyak' (coeliac). The translation of the second line regarding 'miscarriage' was conveyed as 'Pregnancy leads to cognitive issues in women'. The back-translation of the third line would be 'Doctors decided to vaporise the patient'.

8.3.1 Team

Guillem Ramírez from University of Edinburgh took up the task of amending updated Turkish models to allow them to have an additional terminology list.

Sevi Sariisik Tokalac, a professional translator and journalist leading the GoURMET work at BBC, worked on providing the terminology list.

8.3.2 Terminology integration options considered

On 22/11/21 the consortium members met to select the method for building the models. The four options suggested were:

1. **Constrained decoding** (Hokamp, 2017) – beam search is modified to include lexical constraints.
2. **Fine tuning on synthetic data** – build a synthetic parallel corpus out of the terminology words, then fine tune the model.
3. **Soft constraints method** (Dinu, 2019) – the model learns when a constraint is applied.
4. **SYSTRAN placeholder method** (Michon et al, 2020) – dictionary terms are substituted by placeholders in the text. The model outputs a sentence containing the transformed placeholders, which are then converted to the target words.

It was decided to pursue the soft constraints model, based on a custom dictionary added on the docker image of relevant language.

The following terms were then agreed for the work:

1. The work would be conducted on the basis of soft constraints.
2. There would not be a need for data dumps, but only a terminology list in both directions.
3. A small test set would be provided to evaluate it.
4. There would not be human evaluations, but the model would be integrated into a prototype that would be demonstrable.

The benefits of the proposed solution for BBC and DW was that if this model proved workable:

- The list could be regularly updated and expanded as required
- It could pave the way for other domain terminology lists to be included to boost capabilities in the future.

8.3.3 Data and terms

The data and terms were compiled as follows:

Sevi Sariisik Tokalac ran a crawl of stories that appear in the English and Turkish websites about common medical conditions (e.g. cancer, malaria, dementia, respiratory diseases, cardiovascular diseases, neurodegenerative diseases, etc.) and at sight of particular sentences involving medical terms, copied and ran those sentences through the GoURMET translation engine.

Those that were translated correctly were filtered out and those that came up with issues were noted down for further training.

The sentences which included errors or hallucinations and their correct translations were recorded for further re-evaluations.

Meanwhile, common disease names and definitions were compiled from the NHS 'Health A to Z' guide [2], Turkish Ministry of Health 'What is the Disease?' [3], Turkish media outlets, and websites of reputable private healthcare institutions (e.g. Acibadem, Medical Park).

In total, over 800 words were compiled in each direction of the language pairs. These were not always identical. Terms that were commonly used in daily language were preferred over highly medical terms. From this list, 250 sample sentences were compiled in both directions, which was later further expanded in test stage with an additional 50 sentences.

8.3.4 Development methodology

We added a feature that allows users to upload their own dictionary, which could be used to guide the translations in specific domains.

The method implemented was proposed by Dinu et al. (2019) and has been substantially validated in shared tasks studying the usage of terminology (Alam et al., 2021). The core idea behind it is to insert the tranlsation of the term in the source, adding special tokens in the text (<0>, <1>, <2>) marking the beginning and end of the translation of terms. For instance, if our terminology list has the translation pair `indigestion` - `hazımsızlık`, our pre-processing would include the tags:

Several people in the neighbourhood had <0> indigestion <1> hazımsızlık <2>.

The model was then fine tuned to understand that such tags indicate a terminology list pair; hence the model outputs a sentence that contains the target word – the word between the tokens <1> and <2>. This method is based on *soft constraints*, which implies it is possible yet unlikely for the model to not output the target word.

There are two main modifications of the Turkish-ENglish direction: the tagging (pre-processing) and the fine tuning. Besides, we observed that the exclusion of the input token <0> and the source word improves the evaluation metric; therefore, these are excluded at inference time.

The overall process took three months, where methodology and the dictionary were modified given the performance on a sample of parallel sentences in the health domain.

[2] nhs.uk/conditions
[3] sagligim.gov.tr/hastaliklar-durumlar.html

8.3.4.1 Tagging

Tagging refers to the process of looking for words from the terminology list at the input sentence and adding the corresponding tags. For both English and Turkish, the algorithm distinguishes between *proper nouns* and *other words*; however, these two categories are flexible and it is a design question where we should add a new term.

The idea behind this separation is to distinguish between the terms that should be matched to lemmatised words (*other words*) and those that don't accept lemmatised matches (*proper nouns*): if a company is called `Sağlık` (*health*), we don't want to match terms such as `sağlıklı`.

Besides, the agglutinative nature of Turkish and homonyms may cause problems when lemmatising. For instance, the noun `aşılanma` (*vaccine uptake*) can be lemmatised to `aş` (*meal*), `aşı` (*vaccine*), or `aşılan` (*exceed*).

None of these would be satisfactory as we want the lemma to convey a similar meaning to the original word and `aşı` is too generic, meaning that all the derivates from vaccine will collide into the same term. Hence, it may be more sensible to simply treat `aşılanma` as a proper noun.

For proper nouns we follow the same approach in both languages: we check for an exact substring match in the input sentence, taking into account word separators and punctuation marks. For instance, if we considered *Sağlık* a proper noun, we would match '*The company involved in the tender is Sağlık*' or '*The Sağlık company...*' but not the words *Sağlıklı* or *sağlık*.

For other words, we follow a slightly different approach depending on the source language. For English, we lemmatise all the terms in the input sentence and then we look for an exact match. For Turkish we do the same if our term is made of a single word. However, for multiword units of length N, we do an exact match for the first $N-1$ terms – as if they were a proper noun – and we only accept the match if the next word corresponds to the lemmatised N-th term. The reason behind this different approach is that we just want to capture variations in the last term as the previous ones should generally be exactly the same.

We rely on the user to provide their custom terminology, making their own choices on what should be treated as a proper noun. Proper nouns should be indicated in the terminology list with the prefix `PROPN_` followed by the actual word. Our algorithm does not lemmatise neither the input nor the target words in the terminology list, meaning that they should be already lemmatised when provided. Each language direction must have a different dictionary as we want to allow matching multiple words to a single target term.

For English we used spaCy's `en_core_web_sm`, which is one of the most widely used lemmatisers. For Turkish, we used TurkishStemmer (see pypi.org/project/TurkishStemmer). Both lemmatisers are rule-based and have a list of exceptions that could be altered by users in future work.

8.3.4.2 Fine tuning

For the fine tuning of the model, we used the training set from the previous Turkish model. We used the MUSE dictionaries (Conneau et al., 2017) for English-Turkish and looked for sentence pairs that contained a translation pair from the dictionary, adding the corresponding tag only to the source sentence. The training parameters were the same as those used in the pre-training of the model.

8.3.4.3 Results

For the first batch of experiments we wanted to study whether the new model had a different performance than the previous one in general text. We ran the same experiments as in the D5.6 Evaluation Report, using the private test set of 1633 parallel sentences and the evaluation metrics reported in that section. We compared the GoURMET v2 model with the fine tuned model, either using the health terminology list as input or using an empty terminology list.

Tables 14 and 15 show that the performance does not seem to be affected by the fine tuning. There is a slight degradation of the metrics spBLEU and chrF, but the COMET score increases in both directions for the fine tuned model that does not use the terminology list as an input. A user not interested in the health domain should use that model, not leading to a decrease of performance. However, even if that user accidentally uploaded a dictionary, this would not produce a relevant change in the model outputs for the general domain.

We have also evaluated the sentences in the health domain. We have used the test set of 250 sentences described in section 8.3.3. Tables 16 and 17 contain the results.

Table 14: Evaluation in the direction en-tr in the general domain

en-tr	spBLEU	chrF	COMET
GoURMET v2	32.3	52.6	0.91
GoURMET v2 + fine tuned + dictionary	31.7	52	0.90
GoURMET v2 + fine tuned	32	52.1	0.92

Table 15: Evaluation in the direction tr-en in the general domain

tr-en	spBLEU	chrF	COMET
GoURMET v2	37.5	60	0.72
GoURMET v2 + fine tuned + dictionary	37.3	60	0.73
GoURMET v2 + fine tuned	37.5	60	0.73

Table 16: Evaluation in the direction en-tr in the health domain

en-tr	spBLEU	chrF	COMET
GoURMET v2	38.0	56.7	0.77
GoURMET v2 + fine tuned + dictionary	36.4	55.7	0.79

Table 17: Evaluation in the direction tr-en in the health domain

tr-en	spBLEU	chrF	COMET
GoURMET v2	37.1	60.6	0.47
GoURMET v2 + fine tuned + dictionary	37.6	60.9	0.54

Table 18: Copying rate in the direction en-tr

en-tr	Copying rate (tgt word in test set)	Copying rate (general)
GoURMET v2	0.72	0.55
GoURMET v2 + fine tuned + dictionary	0.97	0.88
GoURMET v2 + fine tuned	0.71	0.54

Table 19: Copying rate in the direction tr-en

tr-en	Copying rate (tgt word in test set)	Copying rate (general)
GoURMET v2	0.79	0.63
GoURMET v2 + fine tuned + dictionary	0.98	0.88
GoURMET v2 + fine tuned	0.79	0.63

In the direction English to Turkish, we observe some degradation in spBLEU scores when using the dictionary. However, the COMET score slightly increases with the new model. In the direction Turkish to English, the COMET score increases with the new models and the other metrics remain almost constant. Table 20 shows two samples demonstrating the improvement achieved.

Table 20: Sample translations showing improvements

Original	**Turkish v2**	**Turkish v2+**	**Human**
Germany's government is not entirely united when it comes to rolling out a **vaccine mandate**.	Almanya hükümeti **aşı görevlendirme** konusunda tamamen birleşmiş değil.	Almanya hükümeti **aşı zorunluluğu** konusunda tam olarak birleşmiş değil.	Almanya hükümeti **aşıları zorunlu kılmak** konusunda tam bir fikir birliği içinde değil.
Bedwetting can be the sign of stress in children.	**Yatıştırma** çocuklarda stresin belirtisi olabilir.	**Yatak ıslatma** çocuklarda stres belirtisi olabilir.	**Yatağını ıslatma** çocuklarda stres belirtisi olabilir.

In the first example, the term **vaccine mandate** was conveyed successfully in Turkish V2+terminology list, whereas the earlier form referred to **vaccine assignment** not vaccine mandate. In the second sample, **bedwetting** was initially translated as **assuagement** (mollification) by Turkish V2 probably because **yatıştırma** shares the same first syllable as **yatak**. Adding the terminology list in this case has provided the expected outcome.

There are some limitations with this evaluation. First of all, the test set is very limited – to reach relevant conclusions we should need at least a four times bigger dataset-. In addition, this dataset was a compilation of health parallel sentences without a specific design for our task: in terminology shared tasks (Alam et al., 2021) the terminology terms are commonly tagged and there is an overlap with the terminology list: if a word from the terminology list appears in the source word, the corresponding translation should appear in the target sentence as well.

To account for these limitations, we define the *copying rate*: given that we would tag a sentence by inserting a target word, what is the probability that the new sentence produced contains that target word? Intuitively, for models that don't contain tags (GoURMET v2, GoURMET v2+ fine tuned) this is the percentage of tags that are unnecessary, as we would be guessing them anyway. This number is between 63% and 55% for each direction (Tables 18 and 19). The fine tuned model that uses the terminology list outputs the tag word 88% of the times.

If we restrict the copying rate to those tags whose target word appears in the test set (tgt word in test set), we see that we copy the word almost all the times with the new system. We hypothesise that the model is more reliable in producing a translation that contains the target word when this tag improves the translation, whereas it may ignore the tag at a low rate. We also see that for the other systems this rate is between 70% and 80%, which indicates that the previous models already contain the correct translation most of the times and justify the limited improvements on this test set for automatic evaluation in terms of spBLEU scores.

In conclusion, we have applied a method to improve translations from the health domain. This method is simple, fast and allows the user some control over translations through the terminology list, which is especially interesting for news articles as there are some new topics from time to time and tailoring a terminology list for that topic is a reasonable effort.

The new model effectively copies the tags it sees, and produces better sentences in the health domain. However, our method has some limitations. First of all, homonyms are being given the same tag, which discourages using the terminology list in a general domain. Instead, the terminology list should only be used for news articles in the health domain. However, we have not observed a substantial drop in performance when we use the terminology list in the general domain.

We have also learnt important lessons from using the method of soft constraints. The tagging of the input sentence is often overlooked in the publications that discuss our method. However, for agglutinative languages such as Turkish, this poses a real challenge. The main difficulty of our task has been here, which may complicate the adaptation of this method to other low-resource languages. Another important learning is that evaluating this method is hard and requires a tailored dataset, that contains exactly the term pairs from the terminology list that we want to use.

8.4 Iterations on other languages

Based on earlier learnings, the Amharic model was redone (Lina Murady, Amsterdam) together with Tigrinya.

Having spotted some caching issues with output, the researchers revised the dockers and they were redeployed for Burmese, Pashto, Macedonian.

We learned from these iterations that speed and scalability are the two top requirements to detail in acceptance criteria for future projects. We expand on this point further in our Conclusions.

9 Research Outputs

9.1 Publications

The following publications related to integration work:

See the openAIRE GoURMET page for further details on these publications.

1. Surprise Language Challenge: Developing a Neural Machine Translation System between Pashto and English in Two Months. Alexandra Birch, Barry Haddow, Antonio Valerio Miceli-Barone, Jindřich Helcl, Jonas Waldendorf, Felipe Sánchez-Martínez, Mikel L Forcada, Víctor Sánchez-Cartagena, Juan Antonio Pérez-Ortiz, Miquel Esplà-Gomis, Wilker Aziz, Lina Murady, Sevi Sariisik, Peggy van der Kreeft, Kay Macquarrie. MTSummit 2021, link

2. The University of Edinburgh's English-German and English-Hausa Submissions to the WMT21 News Translation Task. Pinzhen Chen, Jindřich Helcl, Ulrich Germann, Laurie Burchell, Nikolay Bogoychev, Antonio Valerio Miceli Barone, Jonas Waldendorf, Alexandra Birch, Kenneth Heafield. WMT 2021 link

3. The University of Edinburgh's English-Tamil and English-Inuktitut Submissions to the WMT20 News Translation Task. Rachel Bawden, Alexandra Birch, Radina Dobreva, Arturo Oncevay, Antonio Valerio Miceli Barone, Philip Williams. WMT 2020 link

4. GoURMET (Global Under-Resourced Media Translation): Translating Low-Resourced Languages for the Global News Media, poster presentation by Sevi Sariisik Tokalac at Languages & The Media, 13th International Conference and Exhibition on Language Transfer in Audiovisual Media, 21-24 September 2021. link

5. GoURMET - Machine Translation for Low-Resourced Languages: project paper and poster presentation at EAMT, European Association for Machine Translation, by Peggy van der Kreeft, DW, Ghent, 1-3 June 2022. link

9.2 Datasets

The training and test data sets curated by DW and BBC are covered in detail in deliverable D1.4 Final progress report on data gathering and augmentation.

The GoURMET project has released the translation models as open source. The repository contains information about the models, as well as sample code showing how the models can be used. (see github.com/EdinburghNLP/gourmet-models for further details.

10 Conclusions

BBC News Labs has developed three demonstrators to satisfy the purposes set forth at the outset of the GoURMET project, deploying nine language pairs across three prototypes, as well as deploying all 17 models in the public UI. DW has also integrated all 17 models in three prototypes. As a result, both partners have gained a range of insights that will inform future approaches and practices in the field of machine translation in media.

The work has also achieved the benefits envisaged at the outset of the project to a large extent in the face of severe challenges in resourcing due to the *force majeure* of the Covid-19 pandemic that has stretched teams beyond measure and hampered in-person gatherings and communications. We provide a short summary of our findings below:

10.1 Technical insights

The experience demonstrates the need to comprehensively consider the following in any future scoping and deployment plans. Large media organisations such as BBC and DW employ a range of systems and tools which may not necessarily be aligned in terms of the specifications.

In terms of the back end platform and APIs key requirements:

- Before building the API, benchmarking the amount of traffic that will be expected and the use cases of the API
- Understanding the options and feasibility of vertical and horizontal scaling
- Fully examining and better understanding the limitations of cloud services
- Considering other cloud providers, such as Google Cloud that provides a similar service to AWS Fargate but with image caching

- Exploring DevOps options available for machine learning (e.g. Amazon SageMaker, Google Cloud Run)
- Recruiting or enlisting in-house machine learning expertise, which would also allow future explorations in retraining
- Bridging the gap between the research goals and application goals – we found that research goals aimed to focus on quality with less consideration of how the model will interface with end users, where speed and scalability are both essential for utilising the models.

We conclude that back-end and front-end teams need to work together more closely, breaking down any potential silos around work packages.

For example, during the investigations into the Swahili caching issue, neither the BBC nor the developers of the model managed to identify the issue leading to mismatched outputs from requests. It was not until a new software engineer with machine learning experience joined the BBC team and found a few bugs in the models themselves, where previously each model had been treated as a black box, which would only need slotting into a demonstrator. Having skills to bridge the gap, and in this case dig into the model itself, proved extremely useful.

10.2 Usability insights

In terms of demonstrators and prototypes, a significant issue has been the gulf between what users need and what a research project can offer. Throughout the project, a lot of time and effort was invested in managing expectations. When approached for feedback on the output of the exploratory models, end-users have said they expect to interact with a product offering:

- Quality
- Speed
- No reputational risk
- No additional work
- Refined language and style (e.g. DW or BBC-compliant)
- Fully aligned with tooling and visual styling

Essentially users want a mature, finished product, which is out of the scope of what GoURMET is devised to provide. We note that journalists are happy to use the technology in an informal capacity as part of their daily routine (e.g. to learn more about stories related to field of work) but are reluctant to introduce it into their formal workflow (e.g. to trust it to create audience facing output in semi-automated pipelines) and will remain so until the points above are resolved.

What we have learned from this is that we need to improve the following:

- Ensuring future research projects align clearly with business strategy
- Managing expectations of what a research demonstrator or prototype can deliver

- Ensuring there are sufficient resources to conduct the work (e.g. that users can allocate resources to engage with tools when they are not mature – some extra time may be needed initially)
- Securing editorial buy-in from the conception phase
- Managing mistrust and resistance – the aim (at least for the public service BBC and DW) is not to cut costs or replace journalists but to improve the breadth and depth of content available

10.3 Benefits realisation

In deliverable D5.2 Use Cases and Requirements, the use cases for both participating broadcasters address the proposed and potential benefits of the GoURMET project. The findings are described below.

In D5.2 section 3.2.1 BBC Use Cases, five key areas were identified where the MT models developed through GoURMET could make a difference:

A. Improving internal visibility

B. Increased workflow efficiency for reversioning output

C. Editorial oversight

D. Media insight

E. Research and experimentation with semi-automated content production

The work conducted between months 19-42 of the project explored each of these with the relevant stakeholders, and where there was a verified business case, developed the systems and workflows required to achieve the goals.

Table 21 summarises how the BBC addressed four of the five more specific BBC use cases or benefits through the three prototypes.

A. Improving internal visibility

Live Pages Translation has for the very first time enabled the BBC to aggregate and monitor its entire portfolio of output in a language-agnostic setting. Lifting the language barriers, as also demonstrated subsequently in Frank, democratises access of all journalists to all content, and provides a starting point for onward journeys. Despite the lack of official endorsement at managerial levels, or the absence of open roadshows that were initially proposed but failed to materialise due to Covid-19 measures, the solutions we developed have still found their way to users (such as the Serbian, Swahili or Hausa teams) who have established habitual use patterns organically.

B. Increased workflow efficiency for reversioning output

Provided the users' 'high quality' expectations can be satisfied by the model, machine translations help with efficiency. For example, we were told in an interview with the production deputy leader of Serbian service on 26 October 2021 that when using Frank, their translations took as little as a third of the previous time they spent on reversions.

Table 21: BBC use cases

BBC use case	1. Monitoring (LPT)	2. Content creation (Frank)	3. Domain specific (GST)
A. Improving internal visibility	✓	✓	✓
B. Increased workflow efficiency for reversioning output	✓	✓	✓
C. Editorial oversight	✓	✓	✓
D. Media insight	×	×	×
E. Research and experimentation with semi-automated content production	✓	✓	✓
See section	6.1	6.2	6.3

C. Editorial oversight

As the flip side of editorial risk-aversion, providing tools for editorial oversight has proven to be a major opportunity area.

The BBC is organised around hubs – regional units comprising several language services located in cultural, social, and/or physical proximity (e.g. the West Africa Hub, India Hub, Europe Hub). Hub editors can often speak only a couple of the languages they are overseeing. Therefore, when we proposed Frank, some hub editors, particularly those for East Africa and Asia, were keenly interested in being able to have a closer, unmediated understanding of the full range and treatment of day-to-day coverage.

Despite misgivings about whether quality of machine translation is sufficient for audience-facing use, and calls to limit access of journalists to the tools on the basis of manual authorisations, editorial gatekeepers did acknowledge the benefits of providing a bird's-eye view of content to editorial leaders and have agreed to provide unmitigated access so that those with oversight functions can direct their teams to content which can be of use.

D. Media insight

The BBC's focus has shifted away from media insight goals for three reasons:

Firstly, there are issues with copyright and other intellectual property (IP) rights. The primary potential end user in this use case is BBC Monitoring, who are semi-commercial in their operation. There were initial explorations to enable Frank's infrastructure to input external (e.g. non-BBC) content. This was going to be displayed in a custom tab with relevant search and filter functions. However, there were business development and legal concerns that generating a funnel to automate and process large volumes of content from external global media competitors could give the end-product an undue competitive advantage.

Secondly, since the interim report and the changeover of the team, News Labs has pivoted to forge closer relations with BBC World Service Language teams and their editorial needs, while BBC

Monitoring underwent a restructuring process. Therefore, we prioritised facilitating BBC World Service journalists' workflows.

Thirdly, the languages originally proposed for this purpose (Kurdish and Korean) were already served well by the commercial providers by the time the second half of the project began. Since the BBC did not generate content in either of these languages (BBC has Korean output but not North Korean) and due to the scarcity of general media resources around these languages, the partners did not expect to generate considerable benefits from further explorations of these languages.

E. Research and experimentation with semi-automated content production

Content creation was the second major use case we were working towards. We believe, the work has ascertained the two core areas of interest for reversioned content which have divergent sets of requirements:

1. Stories that are generated primarily in English that are on fast developing events/on issues of interest to the target audience but need a quick turnaround to be useful. These assume the existence of high quality models.

2. Stories that draw more engagement, are mainly on human interest topics and are less time sensitive. These assume the models are *not* competent and need significant post-editing to sound 'natural' in the target language.

LPT was designed to cater for the first case and Frank the latter case. Additionally, Frank was conceived as the starting point for a longer term solution which could expand through semi-automated verification, validation, recommendation and curation. Meanwhile, GST addressed the field referred to in D5.2 p21, where *'A new model, trainable by journalists able to create custom dictionaries, could perhaps rapidly improve on previous results'* particularly for *'specialist terminology and proper nouns'* as well as providing a semi-automated pipeline for the visual side of the output.

The three prototypes together formed a multilingual journalism suite, which was shortlisted for BBC News Awards in 2021 for Outstanding Digital Innovation in the BBC.

While there was a general acknowledgement from everyone who encountered the project that the technology is exciting, continuously developing, and offers opportunities, we have also come across suspicion, mistrust and resistance from sections of the editorial ranks in every trial and sampling task we have conducted. This is especially the case with solutions that might reduce the need for direct human involvement, even though any final publications would always be subject to editorial approvals.

When people were asked to justify their resistance, accuracy was raised as the primary concern. Digging into the specifics, some editorial gatekeepers were highly concerned that making content available and transparent for all to see and use might inadvertently boost the rate of inaccuracies and amplify them across the board due to a combination of human and machine errors.

A key editorial gatekeeper in the BBC World Service was so adamant that access to the solutions be curbed, the project team have effectively been barred from seeking direct feedback from journalists in the last five months of the project.

This has severely impacted the team's plans to gather post-edits for Gold Evaluations and proposals to set up a translation validation pipeline in Frank to ease the communication between teams to check ambiguities in translations and promote 'fully validated' translations and 'original journalism' pieces.

In D5.2 section 3.2.2 two broader DW Use Cases were identified:

F. Translation and Adaptation for Content Creation

G. Translation for Cross-Lingual Media Monitoring

During this reporting period (months 19-42 of the project), DW's work focused on integrating the GoURMET models with the language technology systems we have been working on, and then sharing this technnology and the models with our users.

F. Translation and Adaptation for Content Creation

DW has decided that automated language technologies are an essential part of our operations and have embraced their implementation and integration over the past three years. Thus, language technologies, such as automated transcription and translation, will have a major impact on our editorial workflow in particular. In order to guarantee publishable quality, it was decided to opt for a system of automation with full editorial control every step of the way.

Therefore, a system was developed, integrated and implemented that supports the editorial content creation workflow with semi-automated processes, but ultimately leaves the end control and decision with the editor:

plain X within DW and commercial platform

plain X allows early adopters to go full speed on making use of every aspect of automation and providing feedback for enhancements, but also ensures that editors are still in control. Those still hesitant to use it can gradually become familiar with the automation tools.

The need for manual intervention (e.g. post-editing) largely depends on the language pair, and low-resource languages undoubtedly need more attention than high-resourced ones. Also the type and complexity of the content are a factor on the effectiveness of automated translation.

One goal is efficiency in editorial workflows – the need to produce more in less time without adding to the workforce – and this automated process supports that. Another goal is the possibility to expand coverage to other languages. In-depth testing has proven that these objectives are met through automated machine translation and related technologies.

DW is a co-developer of plain X and has decided to implement the toolbox in full to support automated transcription, translation, subtitling and voice-over, with full editorial control in all these processes. In order to make it usable and efficient in the production process, editorial collaboration and review processes are integrated.

The system includes a large number of tools, services and service providers, enabling the selection of the most appropriate service for the content at hand. GoURMET engines have been added as part of the service offer. This allows us to continue to assess its usefulness and compare its output with other engines by analysing the actual use of the models. This brings the output of GoURMET translations to a next level (e.g. publish it as translated subtitles or even as a basis for voice-over).

Even those GoURMET models for which better alternatives are available (i.e. with better MT output) have an added value, as they allow for total control in terms of deployment and do not make use of 'external' service systems such as Google or Microsoft. Thus, they can be used for more sensitive data that needs to stay in-house. Since these models can be installed locally, they can also be used as a low-cost or high-volume engine.

SELMA OSS

GoURMET models have also been successfully integrated in the SELMA Open Source System (OSS), offering an open source workflow for transcription, translation and voice-over for a large number of languages. This serves as a perfect demonstrator for wider use of such language technologies, and means that external users can use the GoURMET models via this tool. It also allows for post-editing of the output text. This makes such processes and technologies available to the open source community, and because of its easy interface in particular also to end users.

This also turns it into a perfect assessment platform for open source systems as one can easily compare the output of one service with another.

Benchmarking

As we aim at the best quality output for language technologies, including MT, we established that a continuous benchmarking process is needed. We have set up a system that allows us to combine automated assessment with user evaluation. The automation of this process enables us to make it sustainable and facilitates a feasible updated assessment in case of enhanced or new engines.

G. Translation for Cross-Lingual Media Monitoring

As part of the SELMA OSS, GoURMET models are also part of a larger multilingual platform where the NLP modules have been integrated to perform in-depth analyses, including Named Entity Recognition and Linking, Summarisation, and Clustering.

These analyses are used to extract and analyse data for reporting or comprehension. The tools do not need to be perfect, but good enough to produce reliable output, which most GoURMET models do, according to our findings. This allows us to include also low-resource languages (and even add some new ones) in our media monitoring.

With low-cost, open source engines, we can process data for comprehension or monitoring purposes. One factor to consider is that locally installed engines used for large datasets require quite a lot of computing power. The alternative of using cloud services also increase costs. Thus, cost, computing power, data security are factors to be considered for media monitoring.

10.4 Summary and recommendations

Overall, regarding the integration and exploitation of the GoURMET outputs, the media partners BBC and DW have duly carried out their commitments as originally proposed in the Grant Agreement (825299) particularly under Tasks 5.3, 5.4 and 5.5 within the scope of Work Package 5.

As noted previously, should the media partners decide to pursue development and deployment of other custom models as a longer term strategy, it would be advisable to ensure the relevant teams have MT and ML expertise on board. Our experience suggests it would also be beneficial to explore working with multiple domain specific terminologies and model retraining options, as well as more detailed criteria around the size and specifications of translation models in future explorations in the field.

Further details of how the learnings, outcomes and benefits will be utilised in the medium to long term can be found in the deliverable D7.4 Sustainability Report .

References

David Adelani, Dana Ruiter, Jesujoba Alabi, Damilola Adebonojo, Adesina Ayeni, Mofe Adeyemi, Ayodele Esther Awokoya, and Cristina España-Bonet. The effect of domain and diacritics in Yoruba–English neural machine translation. In *Proceedings of the 18th Biennial Machine Translation Summit (Volume 1: Research Track)*, pages 61–75, Virtual, August 2021. Association for Machine Translation in the Americas. URL https://aclanthology.org/2021.mtsummit-research.6.

Md Mahfuz Ibn Alam, Ivana Kvapilíková, Antonios Anastasopoulos, Laurent Besacier, Georgiana Dinu, Marcello Federico, Matthias Gallé, Kweonwoo Jung, Philipp Koehn, and Vassilina Nikoulina. Findings of the WMT shared task on machine translation using terminologies. In *Proceedings of the Sixth Conference on Machine Translation*, pages 652–663, Online, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.wmt-1.69.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Maximiliana Behnke, Nikolay Bogoychev, Alham Fikri Aji, Kenneth Heafield, Graeme Nail, Qianqian Zhu, Svetlana Tchistiakova, Jelmer Van der Linde, Pinzhen Chen, Sidharth Kashyap, et al. Efficient machine translation with model pruning and quantization. In *Proceedings of the Sixth Conference on Machine Translation*, pages 775–780, 2021.

Alexandra Birch, Barry Haddow, Antonio Valerio Miceli Barone, Jindrich Helcl, Jonas Waldendorf, Felipe Sánchez Martínez, Mikel Forcada, Víctor Sánchez Cartagena, Juan Antonio Pérez-Ortiz, Miquel Esplà-Gomis, Wilker Aziz, Lina Murady, Sevi Sariisik, Peggy van der Kreeft, and Kay Macquarrie. Surprise language challenge: Developing a neural machine translation system between Pashto and English in two months. In *Proceedings of Machine Translation Summit XVIII: Research Track*, pages 92–102, Virtual, August 2021. Association for Machine Translation in the Americas. URL https://aclanthology.org/2021.mtsummit-research.8.

O. Bojar, C. Federmann, M. Fishel, Y. Graham, B. Haddow, P. Koehn, and C. Monz. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6401. URL https://www.aclweb.org/anthology/W18-6401.

Peng-Jen Chen, Jiajun Shen, Matt Le, Vishrav Chaudhary, Ahmed El-Kishky, Guillaume Wenzek, Myle Ott, and Marc'Aurelio Ranzato. Facebook ai's wat19 myanmar-english translation task submission. *arXiv preprint arXiv:1910.06848*, 2019.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July

2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://aclanthology.org/2020.acl-main.747.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1294. URL https://aclanthology.org/P19-1294.

Miquel Espla-Gomis and Mikel Forcada. Combining content-based and url-based heuristics to harvest aligned bitexts from multilingual sites with bitextor. *The Prague Bulletin of Mathematical Linguistics*, 93(1):77–86, 2010.

Ignatius Ezeani, Paul Rayson, Ikechukwu Onyenwe, Chinedu Uchechukwu, and Mark Hepple. Igbo-english machine translation: An evaluation benchmark, 2020. URL https://arxiv.org/abs/2004.00648.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Michael Auli, and Armand Joulin. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48, 2021. URL http://jmlr.org/papers/v22/20-1307.html.

Barry Haddow and Faheem Kirefu. Pmindia–a collection of parallel corpora of languages of india. *arXiv preprint arXiv:2001.09907*, 2020.

Barry Haddow, Nikolay Bogoychev, Denis Emelin, Ulrich Germann, Roman Grundkiewicz, Kenneth Heafield, Antonio Valerio Miceli Barone, and Rico Sennrich. The University of Edinburgh's submissions to the WMT18 news translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 399–409, Belgium, Brussels, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6412. URL https://aclanthology.org/W18-6412.

V.C.D. Hoang, P. Koehn, G. Haffari, and T. Cohn. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia, July 2018. URL https://www.aclweb.org/anthology/W18-2703.

Bushra Jawaid and Daniel Zeman. Word-order issues in english-to-urdu statistical machine translation. *Prague Bull. Math. Linguistics*, 95:87–106, 2011.

Bushra Jawaid, Amir Kamran, and Ondřej Bojar. Urdu monolingual corpus, 2014. URL http://hdl.handle.net/11858/00-097C-0000-0023-65A9-5. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Girish Nath Jha. The tdil program and the indian langauge corpora intitiative (ilci). In *LREC*, 2010.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017. doi: 10.1162/tacl_a_00065. URL https://doi.org/10.1162/tacl_a_00065.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, et al. Marian: Fast neural machine translation in c++. *arXiv preprint arXiv:1804.00344*, 2018.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL https://aclanthology.org/D16-1139.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Tom Kocmi and Ondřej Bojar. Trivial transfer learning for low-resource neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 244–252, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6325. URL https://aclanthology.org/W18-6325.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Association for Computational Linguistics Companion Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P07/P07-2045.

Philipp Koehn, Vishrav Chaudhary, Ahmed El-Kishky, Naman Goyal, Peng-Jen Chen, and Francisco Guzmán. Findings of the WMT 2020 shared task on parallel corpus filtering and alignment. In *Proceedings of the Fifth Conference on Machine Translation*, pages 726–742. Association for Computational Linguistics, 2020.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012.

R.V. Lim, K. Heafield, H. Hoang, M. Briers, and A.D. Malony. Exploring hyper-parameter optimization for neural machine translation on GPU architectures. *CoRR*, abs/1805.02094, 2018. URL http://arxiv.org/abs/1805.02094.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020. doi: 10.1162/tacl_a_00343. URL https://aclanthology.org/2020.tacl-1.47.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. 1993.

Anthony McEnery, Paul Baker, Robert Gaizauskas, and Hamish Cunningham. Emille: Building a corpus of south asian languages. In *Proceedings of the International Conference on Machine Translation and Multilingual Applications in the new Millennium: MT 2000*, 2000.

Robert C Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, 2010.

Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014. URL https://github.com/fmfn/BayesianOptimization.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, ACL'02, pages 311–318, Philadelphia, Pennsylvania, USA, 2002.

Jerin Philip, Shashank Siripragada, Vinay P Namboodiri, and CV Jawahar. Revisiting low resource status of indian languages in machine translation. In *8th ACM IKDD CODS and 26th COMAD*, pages 178–187. 2021.

Maja Popović. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618, 2017.

Matt Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6319.

Matt Post, Chris Callison-Burch, and Miles Osborne. Constructing parallel corpora for six indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, 2012.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. The curious case of hallucinations in neural machine translation. *CoRR*, abs/2104.06683, 2021. URL https://arxiv.org/abs/2104.06683.

Víctor M. Sánchez-Cartagena, Marta Bañón, Sergio Ortiz-Rojas, and Gema Ramírez. Prompsit's submission to WMT 2018 parallel corpus filtering shared task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 955–962, Belgium, Brussels, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6488. URL https://www.aclweb.org/anthology/W18-6488.

Víctor M. Sánchez-Cartagena, Miquel Esplà-Gomis, Juan Antonio Pérez-Ortiz, and Felipe Sánchez-Martínez. Rethinking data augmentation for low-resource neural machine translation: A multi-task learning approach. In *Proceedings of the 2021 Conference on Empirical Methods*

in Natural Language Processing, pages 8502–8516, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.669. URL https://aclanthology.org/2021.emnlp-main.669.

Holger Schwenk, Marta R. Costa-jussà, and Jose A. R. Fonollosa. Smooth bilingual *n*-gram translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 430–438, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/D07-1045.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6490–6500, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.507. URL https://aclanthology.org/2021.acl-long.507.

R. Sennrich, B. Haddow, and A. Birch. Edinburgh Neural Machine Translation Systems for WMT 16. In *Proceedings of the First Conference on Machine Translation*, pages 371–376, Berlin, Germany, August 2016a. URL http://www.aclweb.org/anthology/W/W16/W16-2323.

R. Sennrich, A. Birch, A. Currey, U. Germann, B. Haddow, K. Heafield, A.V. Miceli Barone, and P. Williams. The University of Edinburgh's Neural MT Systems for WMT17. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 389–399, Copenhagen, Denmark, September 2017. URL http://www.aclweb.org/anthology/W17-4739.

Rico Sennrich and Biao Zhang. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1021. URL https://www.aclweb.org/anthology/P19-1021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California, June 2016b. Association for Computational Linguistics. doi: 10.18653/v1/N16-1005. URL https://aclanthology.org/N16-1005.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany, August 2016c. Association for Computational Linguistics. doi: 10.18653/v1/P16-1009. URL https://www.aclweb.org/anthology/P16-1009.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, ACL'16, pages 1715–1725, Berlin, Germany, 2016d.

Shashank Siripragada, Jerin Philip, Vinay P. Namboodiri, and C V Jawahar. A multilingual parallel corpora collection effort for Indian languages. In *Proceedings of the 12th Language Resources*

and Evaluation Conference, pages 3743–3751, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.462.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*, 2020.

Solomon Teferra Abate, Michael Melese, Martha Yifiru Tachbelie, Million Meshesha, Solomon Atinafu, Wondwossen Mulugeta, Yaregal Assabie, Hafte Abera, Binyam Ephrem, Tewodros Abebe, Wondimagegnhue Tsegaye, Amanuel Lemma, Tsegaye Andargie, and Seifedin Shifaw. Parallel corpora for bi-directional statistical machine translation for seven Ethiopian language pairs. In *Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing*, pages 83–90, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/W18-3812.

Jörg Tiedemann. The tatoeba translation challenge – realistic data sets for low resource and multilingual MT. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1174–1182, Online, November 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.wmt-1.139.

Jörg Tiedemann and Santhosh Thottingal. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal, November 2020. European Association for Machine Translation. URL https://aclanthology.org/2020.eamt-1.61.

Jörg Tiedemann. Parallel data, tools and interfaces in opus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7.

Chau Tran, Shruti Bhosale, James Cross, Philipp Koehn, Sergey Edunov, and Angela Fan. Facebook ai wmt21 news translation task submission. *arXiv preprint arXiv:2108.03265*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017a. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017b.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.494.

Appendix A Translation Model Details

This section describes the specifics of the translation models as delivered for integration. We describe the models which either translate into, or out of English. In particular, the descriptions will cover most of these items:

- Bilingual data used.
- Monolingual data used (for instance, for back-translation (Sennrich et al., 2016c)).
- Language resources used (for instance, part-of-speech taggers, morphological analysers, bilingual dictionaries, machine translation systems etc.)
- NMT platform and architecture
- Training specifics (multi-source input, back-translation, integration of language resources, etc.).
- Indicators of the quality or usefulness of their output.

Automatic evaluation results are reported below for the models as they have been developed by the researchers. We have further, extensive evaluation of these models described in deliverable 5.6. Final Progress Report on Evaluation.

The data for training models is available to download as detailed in deliverables D1.3 Initial Release of Project Data (for the first set of languages) and D1.5 Final Release of Project Data (for the second set of languages).

A.1 Swahili↔English

See D5.3 section A.1

A.2 Gujarati↔English

See D5.3 section A.2

A.3 Turkish↔English

See D5.3 section A.3

A.4 Bulgarian↔English

See D5.3 section A.4

A.5 Tamil↔English

See D5.3 section A.5

A.6 Serbian↔English

See D5.3 section A.6

A.7 Amharic↔English

See D5.3 section A.7

A.8 Kyrgyz↔English

See D5.3 section A.8

A.9 Macedonian↔English

This section describes the resources used and the steps followed to build the English–Macedonian NMT systems for both translation directions. A combination of multilingual neural machine translation (Johnson et al., 2017), and back-translation was applied with the aim of leveraging all the available sources of information for this language pair.

A.9.1 Corpora

Tables 22 and 23 show, respectively, the parallel and monolingual corpora used for training the English–Macedonian NMT models.

As regards parallel corpora, all the corpora available from the OPUS[4] website was used together with one additional parallel corpus: the GoURMET corpus, which was crawled from the web following the method described in deliverable D1.4. That method involved identifying and crawling parallel websites from the top-level domain `.mk`, processing them with Bitextor (Espla-Gomis and Forcada, 2010) and filtering the resulting parallel sentences with Bicleaner (Sánchez-Cartagena et al., 2018).

Concerning monolingual corpora, four corpora were used: the NewsCrawl (Bojar et al., 2018) for English, the NewsCrawl for Macedonian,[5] a set of news articles written in Macedonian provided by Deutsche Welle (similar to NewsCrawl), and the GoURMET monolingual corpus for Macedonian. The first three corpora were chosen because they belong to the news domain, the same domain of application of the NMT systems built. Given that the size of the Macedonian monolingual corpora is much smaller than the size of the English monolingual corpus, the GoURMET monolingual corpus for Macedonian, which contains additional monolingual data obtained as a by-product of the process of crawling parallel data from the web, was also used.

All the English–Bulgarian parallel corpora available from the OPUS[6] website, whose details are depicted in Table 24, were also used for training multilingual machine translation systems.

[4] http://opus.nlpl.eu/
[5] http://data.statmt.org/news-crawl/mk/
[6] http://opus.nlpl.eu/

Corpus	sentences	en tokens	mk tokens
GoURMET	54 795	1 690 817	1 535 711
JW300 v1	521 445	8 967 755	8 648 520
Ubuntu v14.10	3 310	15 020	15 428
GNOME v1	117	290	316
QED v2.0a	68 113	958 190	887 458
Tatoeba v20190709	80 284	441 596	412 896
SETIMES	207 777	4 430 556	4 461 862
Global Voices	45 947	868 435	837 145
OpenSubtitles*	3 401 326	21 243 724	18 170 033
Total	4 452 360	38 910 415	35 314 931

Table 22: Parallel English–Macedonian corpora used to train the NMT systems. Corpora flagged with * were aggressively filtered.

Corpus	sentences	tokens
NewsCrawl (en)	32 000 000	642 976 627
NewsCrawl (mk)	370 345	8 017 928
Deutsche Welle (mk)	1 153 565	22 912 891
GoURMET (mk)	869 415	25 527 173

Table 23: Monolingual Macedonian and English corpora used to build synthetic parallel data through back-translation.

Finally, development and test corpora, whose statistics are shown in Table 25, were produced by automatically aligning and manually validating Deutsche Welle news articles.

Preprocesing. Although OpenSubtitles is by far the largest English–Macedonian parallel corpus, it is very noisy and its domain (film subtitles) largely differs from the news domain. Hence, including the whole of it in the training data could harm the translation quality of the final systems. Hence, we filtered it and chose the top 1 million parallel sentences whose English side resemble news data most via cross-entropy data selection (Moore and Lewis, 2010). We used NewsCrawl to build the in-domain English language model and the English side of OpenSubtitles itself to build the out-of-domain language model.

All corpora were tokenized with the Moses tokenizer (Koehn et al., 2007) and truecased. Parallel sentences with more than 100 tokens in either side were removed. Words were split in sub-word units with byte pair encoding (BPE; Sennrich et al. (2016d)). Table 47 reports the size of the corpora after this pre-processing.

Corpus	pair	sentences	SL tokens	TL tokens
Opus	English–Bulgarian	9 826 017	176 928 883	171 633 198

Table 24: Parallel corpora from other language pairs used to train the NMT systems.

Corpus	sentences	en tokens	mk tokens
development	1 000	15 391	14 936
test	1 000	15 197	14 789

Table 25: Development and test corpora.

Corpus	Languages	sentences	SL tokens	TL tokens
parallel	English–Macedonian	2 021 571	33 454 701	36 504 220
NewsCrawl	English	32 000 000	906 086 026	
NewsCrawl	Macedonian	370 345	11 537 057	
GoURMET mono	Macedonian	869 415	48 165 548	
Deutsche Welle	Macedonian	1 153 565	32 740 224	
Opus	English–Bulgarian	9 512 357	221 894 262	235 204 744

Table 26: Size of the corpora used to build the NMT systems after preprocesing. For the English NewsCrawl corpus, only the size of the subset that has been used for training is displayed. Token counts were calculated after BPE splitting with 10 000 operations.

A.9.2 Model architecture and training

The NMT models were trained with the fairseq toolkit (Ott et al., 2019). Since training hyper-parameters can have a large impact in the quality of the resulting system (Lim et al., 2018; Sennrich and Zhang, 2019), a grid search was carried out in order to find the best hyper-parameters for each translation direction. Both the Transformer (Vaswani et al., 2017b) and recurrent neural network (RNN) with attention (Bahdanau et al., 2014) architectures were explored. The starting points were the Transformer hyper-parameters[7] described by Sennrich et al. (2017) and the RNN hyper-parameters[8] described by Sennrich et al. (2016a).

The best set of hyperparameters for the systems which were trained solely on the available parallel data for English–Macedonian were first determined. The following hyperparameters were then explored for each translation direction and architecture:

- Number of BPE operations: 5 000, 10 000, 20 000, 40 000 or 80 000.
- Model size. For RNN systems, we explored the following hidden/embedding size pairs: 1024/512, 512/512 and 256/256. For Transformer systems, we explored the following model sizes: 512, 256, 128.

Afterwards, grid search was repeated for multilingual systems, which were train on the concatenation of the English–Macedonian and English–Bulgarian data depicted in Table 47. Given the training overhead caused by the addition of large amounts of English–Bulgarian data, only the Transformer and RNN starting points mentioned above were compared, using 1 GPU and 20 000 BPE operations in both cases.

In all cases, early stopping was based on perplexity on the development set and patience was set

[7] https://github.com/marian-nmt/marian-examples/tree/master/wmt2017-transformer
[8] https://github.com/marian-nmt/marian-examples/tree/master/training-basics

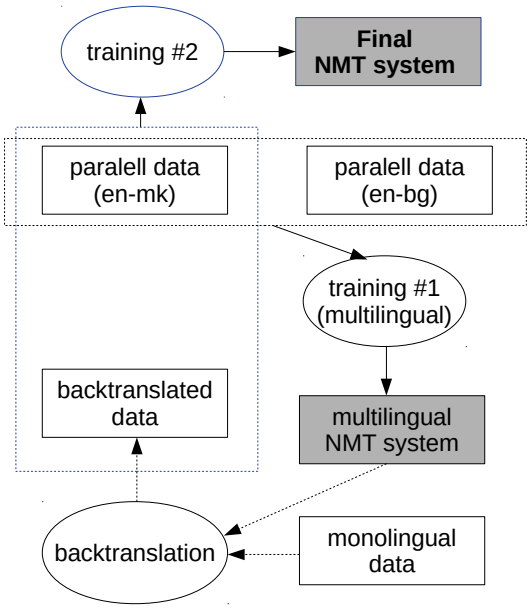

Figure 35: Steps followed to train the final English-to-Macedonian and Macedonian-to-English systems. The final system is highlighted in bold.

to 10 validations, with a validation carried out every 5000 updates. Batch size was set to 5 000 tokens.

For the systems trained only on parallel corpora, the best performing systems for both directions followed the Transformer architecture with the largest model size and 10 000 BPE operations. For multilingual systems, the Transformer architecture was also the most effective one. Multilingual systems outperformed systems trained solely on the English–Macedonian data for both directions. Hence, they were the starting point for the process carried out to leverage monolingual data, which is described next.

Leveraging monolingual data. Systems trained solely on parallel data were improved by making use of monolingual corpora via back-translation. Since the quality of a system trained on back-translated data is usually correlated with the quality of the system that translates the TL monolingual corpus into the SL (Hoang et al., 2018, Sec. 3), the systems were trained by following these steps, which are summarized in Figure 35.

1. Train multilingual systems using a combination of English–Macedonian and English–Bulgarian parallel corpora.
2. Back-translate (Sennrich et al., 2016c) the monolingual data.
3. Train the final systems on all the resources available and go back to step 2.

Steps 2-3 were executed 5 times in total.

For the first step, a different multilingual system was trained for each direction. In other words, two multilingual systems were trained: an English-to-Macedonian plus English-to-Bulgarian system, and a Macedonian-to-English plus Bulgarian-to-English system. In both cases, the systems were

Strategy	BLEU	chrF++
English→Macedonian		
only parallel	43.07	67.88
+ multilingual	44.19	68.60
+ backtranslation	48.42	71.66
Google Translate	49.16	72.04
Macedonian→English		
only parallel	54.98	74.43
+ multilingual	57.31	75.85
+ backtranslation	58.59	77.01
Google Translate	72.63	85.66

Table 27: Automatic evaluation results for English–Macedonian obtained for the different development steps of the MT systems.

trained on the concatenation of the same English–Macedonian and English–Bulgarian parallel data described in Table 47. Corpora were just concatenated, without any kind of oversampling. Systems were fine-tuned on the English–Macedonian data only.

The systems from the first step were used to back-translate the monolingual data described in Table 47. Then, systems that took advantage of the newly back-translated data were trained in step 3.

In the first 2 executions of step 3, systems were multilingual, and they were trained on the concatenation on the genuine parallel data, the back-translated data and the English–Bulgarian data, and fine-tuned on the genuine parallel data. In the remaining executions, since we noticed that the English–Bulgarian data was no longer useful, systems were trained only on the concatenation of the original parallel data and the back-translated data, and fine-tuned on the former.

A.9.3 Indicators of quality

Table 27 shows, for the different steps in the development of the MT systems, the BLEU (Papineni et al., 2002) and chrF++ (Popović, 2017) scores (the latter is multiplied by 100 to improve readability) computed on an in-house test set created by BBC.[9] It is worth noting the positive effect of all forms of additional data leveraged: multilingual NMT, and monolingual data via back-translation.

A.10 Hausa↔English

In this section we describe models trained for English–Hausa and Hausa–English translation. The training of the systems is based on iterative backtranslation (Hoang et al., 2018). Our final systems are obtained with transfer learning from English–German and German–English systems, using fine-tuning on the final version of the backtranslated data (Kocmi and Bojar, 2018).

[9] Note that, since we have no control over the data Google Translate is trained on, there is no guarantee that its training data contains part of the test set.

A.10.1 Corpora

Parallel data for English–Hausa was obtained from OPUS, crawling the web following the methods in D1.4 (the GoURMET corpus), and by using similar parallel data extraction method applied on news articles obtained from Deutsche Welle. Refer to Table 28 for the sizes of the individual parallel corpora.

Corpus		sentences	En tokens	Ha tokens
GoURMET (Internet Archive)	original	72 776	1 118 841	1 093 988
	deduplicated	19 904	417 458	413 871
Deutsche Welle	clean+noisy	10 402	149 745	295 636
	clean	1 104	20 168	36 048
OPUS (Tiedemann and Thottingal, 2020)	JW300	237 065	4 118 848	4 577 117
	Tanzil	128 376	2 375 731	2 392 175
	Tatoeba	56	378	317
Total	original	448 675	7 763 543	8 359 233
	training size	493 317	9 195 121	10 277 482

Table 28: Size of the parallel corpora used to build the English–Hausa NMT systems.

Table 29 shows the sizes of monolingual corpora used for backtranslation. For Hausa, we used sentences crawled from the Internet Archive, documents obtained from Deutsche Welle and News Crawl. For English, we use the 2018 edition of News Crawl.

Corpus	sentences	tokens
Internet Archive (ha)	1 163 513	9 749 042
Deutsche Welle (ha)	212 841	7 340 074
News Crawl (ha)	405 827	9 440 423
Total – Hausa	1 782 181	26 529 539
News Crawl 2018 (en)	18 112 579	395 809 896

Table 29: Size of the monolingual corpora used to build the English–Hausa NMT systems.

All corpora were tokenized with the Moses tokenizer (Koehn et al., 2007) and truecased. Words were split in sub-word units with byte pair encoding (BPE; Sennrich et al. (2016d)).

A.10.2 Model architecture and training

The NMT models were trained using the Marian toolkit (Junczys-Dowmunt et al., 2018). We begin with two models per translation direction. First, we train a translation model only on the data avaiable from OPUS (called "OPUS"). Second, we train another translation model (called "Parallel") on the OPUS data, plus the rest of the cleaned parallel data (See Table 28). We upsample the datasets obtained from the Internet Archive (deduplicated) and from Deutsche Welle (clean)

by factors of 5 and 30 respectively, to better balance the data distribution in favor of cleaner data sources.

We perform 2 rounds of iterative backtranslation with the translations produced by each model used for training of the new model in the other direction. In the first round, we use both "OPUS" and "Parallel" models to create the backtranslations using Internet Archive and Deutsche Welle monolingual data (translated from Hausa to English), and News Crawl 2018 (translated from English to Hausa). These corpora are mixed with the parallel training data for the "Parallel" model in the respective translation direction. In the second round, we update the backtranslations of the monolingual data with the models from the previous iteration, and we include the Hausa News Crawl in Hausa-to-English backtranslation.

After the 2 rounds of backtranslation, we initialized the training of the final models with a pre-trained English–German and German–English models, where we replace the German side of the dataset with Hausa. We employed this technique successfully for Igbo, described in the following section (A.11).

A.10.3 Indicators of quality

We report SacreBLEU scores on internal BBC development and test sets in Table 30.

Model	English–Hausa		Hausa–English	
	dev	test	dev	test
OPUS	9.4	7.3	11.0	8.3
Parallel	7.1	5.5	9.5	7.2
BT #1	14.7	12.1	21.4	16.1
BT #2	20.4	16.2	25.4	18.8
+ Fine-tuning	**22.8**	**17.7**	**28.4**	**20.5**

Table 30: SacreBLEU results for English↔Hausa

A.11 Igbo↔English

A.11.1 Corpora

Parallel data for English–Igbo was obtained from various datasets. We prepared bilingual training of 472,694 sentence pairs collected from the following corpora: GNOME,[10] Internet Archive[11], JW300,[12] Tatoeba,[13] Ubuntu.[14]

The initial corpus is medium-sized and provides a fair coverage of different domains, however it is not necessarily high-quality and in-domain for news translation. We supplemented this corpus

[10] http://opus.nlpl.eu/GNOME.php
[11] Internal crawl
[12] JW300 is no longer publicly available, although a version seems to have been preserved in the IGBONLP repository
[13] https://opus.nlpl.eu/Tatoeba-v2021-03-10.php
[14] http://opus.nlpl.eu/Ubuntu.php

with the smaller (10,000 sentence pairs) but higher quality IGBONLP[15] (Ezeani et al., 2020), which contains translations checked for quality by native speakers.

For backtranslation, we used Igbo monolingual data data from Newscrawl[16] and IGBONLP, for a total of 444,332 sentences and English monolingual data from Newscrawl, for a total of 33,600,797 sentences after cleaning and deduplication.

Corpus		sentences	En tokens	Ig tokens
OPUS (Tiedemann and Thottingal, 2020)	JW300	475,206	8,078,844	10,294,393
	GNOME	23,767	132,007	132,829
	Ubuntu	635	3,422	3,469
	Tatoeba	21	154	150
Internet Archive		25,969	477,042	489,916
IGBONLP (Ezeani et al., 2020)		10,000	176,375	184,538
Total (w/o IGBONLP)	original	525,598	8,691,469	10,920,757
	pre-processed	472,694	7,032,961	9,033,702

Table 31: Size of the parallel corpora used to build the English-Igbo NMT systems.

Language	Corpus	sentences	tokens
Igbo	IGBONLP	383,449	5,724,201
	NewsCrawl2019	42,086	677,311
	NewsCrawl2020	18,797	305,146
	Total	444,332	6,706,658
English	NewsCrawl	33,600,797	836,569,433

Table 32: Monolingual resources used in the English-Igbo NMT systems (filtered and deduplicated).

A.11.2 Model architecture and training

We trained Transformer-base models using the Marian toolkit with multiple phases of pretraining.

We begin with a German↔English model trained on WMT2021 data (constrained news translation task). German is not strongly related to Igbo, ideally it would have been preferable to pretrain on a related high-resource language, however such data does not exist, and multi-lingual pretraining on any natural language is known to improve quality, especially because it helps the model to process and generate English. We fine tune this model on the Igbo-English parallel data (with the high-quality IGBONLP corpus oversampled by a factor of 47) to obtain an initial Igbo→English model.

We then perform iterative backtranslation: we use the initial Igbo→English model to translate the Igbo monolingual corpus, we combine the result with the base parallel corpus to train an

[15] https://github.com/IgnatiusEzeani/IGBONLP/tree/master/ig_en_mt
[16] https://data.statmt.org/news-crawl/

English→Igbo model, which we then fine tune on the IGBONLP corpus. We perform another two rounds of backtranslation and fine tuning, alternating the model direction, obtaining our final models.

A.11.3 Indicators of quality

We evaluate quality of our translation systems by computing the SacreBLEU scores on the public IGBONLP, FLORES-101,[17] test sets and an internal BBC test set. We report the results in table 33.

English→Igbo			Igbo→English		
IGBONLP	FLORES	BBC	IGBONLP	FLORES	BBC
12.6	11.2	14.1	12.8	16.6	19.4

Table 33: SacreBLEU results on test sets for English–Igbo.

A.12 Tigrinya↔English

For English-Tigrinya we investigated two strategies. First, we opted for training a multilingual model in an attempt to benefit from resources gathered for English-Amharic, given that Tigrinya and Amharic are related languages. Despite various settings and an extensive hyperparameter search, this approach did not lead to appreciable improvements compared to the more established approach of training a system using parallel and synthetic data obtained via back-translation (Sennrich et al., 2016c). Hence, we opted for the latter. In an attempt to further improve the quality of our systems, we experimented with iterative backtranslation (Hoang et al., 2018).

Summary of approach:

1. We start from a baseline model trained on the available parallel corpora.
2. Next, we use this system to obtain backtranslations for the available monolingual corpora, creating synthetic translation pairs that can be used for further training the baseline component.
3. We now train on a combination of parallel and synthetic data, and also investigate the effect of fine-tuning the resulting system on the parallel original parallel portion of the data.
4. At this point, we hopefully have a better system, which we can use to obtain improved backtranslations for the monolingual data (i.e., repeat step 2), which in turn can be used to improve the overall system (i.e., repeat step 3).

[17] https://github.com/facebookresearch/flores

A.12.1 Corpora

We gathered parallel resources from three sources, namely, OPUS (Tiedemann and Thottingal, 2020), the Travis Foundation[18] and Teferra Abate et al. (2018)'s parallel corpus of Ethiopian languages. Table 34 lists the resources.

Corpus		sentences	SL tokens	TL tokens
OPUS (Tiedemann and Thottingal, 2020)	JW300	399 452	5 957 330	5 604 130
	Tatoeba	69	426	305
	Wikimedia	5	169	116
TravisFoundation		6 3465	221 3296	173 9082
Teferra Abate et al. (2018)		3 6025	862 484	569 210
Total	original	499 016	10 988 623	7 912 843
	pre-processed	417 232	8 117 440	6 843 353

Table 34: Size of the parallel corpora used to build the English-Tigrinya NMT systems.

Language	Corpus	sentences	tokens
Tigrinya	GouRMET	152 554	2972988
	NewsCrawl2018	28 068	599 991
	NewsCrawl2019	66 822	1 495 053
	NewsCrawl2020	50 289	1 145 932
	Total	437 090	9 001 849
English	NewsCrawl	3 500 000	87 133 067

Table 35: Monolingual resources used in the English-Tigrinya NMT systems.

We preprocessed the corpora to remove duplicates, to remove non-ge'ez script from the Tigrinya side and remove ge'ez script from the English. We use BPE-segmentation with a separate vocabulary of 5000 codes for each language.

In order to develop our systems, we reserve 6,000 sentences from the parallel training data to form a development and a test set (3000 sentences each). In addition, we have access to development (600 sentence pairs) and test (600 sentence pairs) sets based on BBC data.

A.12.2 Model architecture and training

We use fairseq (Ott et al., 2019) to train 5-layer transformer models (Vaswani et al., 2017a) with an embedding dimension of 512. We used the Adam optimizer (Kingma and Ba, 2015) and *inverse sqrt* learning rate scheduler. Hyperparameter search for dropout rate, label smoothing coefficient, weight decay and initial learning rate was performed through Bayesian Optimisation (Snoek et al., 2012).[19] These are the values returned by BayesOpt: dropout rate (0.2), label smoothing coefficient

[18] https://github.com/travisfoundation/Tigrinya-Parallel-Corpus
[19] We use the packaged by Nogueira (2014).

(0.1), weight decay (0.01), intial learning rate (0.001). We performed this search on the parallel portion of the data and kept the hyperparameters fixed for further experiments. However, we found that for backtranslation experiments the learning rate was too high and settled on a smaller learning rate 0.0005, which we also used for the fine-tuning experiments. We apply early stopping with patience of 5 on the BBC dev set and select models on the validation label smoothed loss.

A.12.3 Indicators of quality

Table 36 shows translation results on two dev sets in terms of SacreBLEU (Post, 2018).[20] We can see that iterative backtranslation tends to help, but sometimes it further requires fine tuning the resulting system on parallel data alone (without synthetic back-translated data). Table 37 shows the results on the two test sets for the intial system, trained on parallel data only, and the final system, trained on parallel data and syntehtic data from two rounds of backtranslation. The impact of backtranslation is substantial, as expected. It is also clear that translating into Tigrinya is more challenging than from Tigrinya.

Model	English-Tigrinya		Tigrinya-English	
	heldout	BBC	heldout	BBC
Parallel	27.23	1.41	30.49	4.43
+ backtranslation (iteration 1)	26.02	3.35	29.86	11.20
+ fine tuning on parallel	28.27	2.97	35.60	11.88
+ backtranslation (iteration 2)	25.90	4.91	29.27	11.17
+ fine tuning on parallel	28.13	4.17	35.64	12.06

Table 36: SacreBLEU results on development sets for English–Tigrinya.

Model	English-Tigrinya		Tigrinya-English	
	heldout	BBC	heldout	BBC
Initial	28.01	1.61	31.78	4.99
Final	28.71	5.13	37.01	12.95

Table 37: SacreBLEU results on test sets for English–Tigrinya.

A.13 Pashto↔English

This section describes the resources exploited as well as the steps followed in order to build the English–Pashto NMT systems for both translation directions. Fine-tuning of a large pretrained model with back-translated data was carried out to train our system.

In the media industry, the focus of global reporting can shift overnight. There is a compelling need to be able to develop new machine translation systems in a short period of time, in order to more efficiently cover quickly developing stories. The GoURMET project undertook its *surprise*

[20] Signature: `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0`.

language evaluation as an exercise to bring together the whole consortium to focus on a language pair of particular interest to the media partners for a short period of time. On 1st February 2021, BBC and DW revealed the chosen language to be Pashto. By completing and documenting how this challenge was addressed,[21] we proved we are able to bootstrap a new high quality NMT task within the very limited window of two months.

We developed two different neural models: a *from-scratch* system, and a larger and slower system based on an existing pretrained model. The development of the former starts with a mediun-size randomly-initialized transformer (Vaswani et al., 2017a), whereas the latter is obtained by fine-tuning the larger downloadable mBART50 pretrained system (Tang et al., 2020). As the mBART50-based model gave better results in both automatic and manual evaluation, we will focus hereinafter on this model. The from-scratch system is described in our conference paper (Birch et al., 2021).

A.13.1 Corpora

Traning data consists of English–Pashto parallel data as well as monolingual data, and was obtained by three means: explotation of corpora available online, directly crawling websites likely to contain parallel data, and crawling the top-level domain of Afganistan (domain `.af`), where Pashto is an official language. Table 38 shows the number of segment pairs, the number of tokens both in Pashto and English, and the average number of tokens per segment for the crawled corpora. Statistics for the downloaded corpora will be presented later.

		Pastho		English	
Corpus name	# segm. pairs	# tokens	tokens/segm.	# tokens	tokens/segm.
Crawled	59,512	759,352	12.8	709,630	11.9
BBC Test	1,350	25,453	18.8	30,417	22.5
BBC Dev	1,000	18,793	18.8	22,438	22.4
DW Test	813	14,956	18.3	20,797	25.5
FLORES-101	1,012				

Table 38: Crawled and in-house parallel corpora statistics.

Development and test corpora, whose statistics are also shown in Table 38, were produced by automatically aligning and manually validating BBC and DW news articles. FLORES 101,[22] which is a multilingual translation benchmark dataset for 101 languages, was used as additional evaluation corpora.

We used SentencePiece[23] (Kudo and Richardson, 2018) to split words into subwords. As our models were obtained by fine-tuning the mBART50 pretrained model, we used its own published SentencePiece tokenizer.

[21] Our main conclusions were presented (Birch et al., 2021) in a conference. The discussion here is mainly based on that conference paper.
[22] https://github.com/facebookresearch/flores
[23] https://github.com/google/sentencepiece

A.13.2 Model architecture and training

Our systems are based on the pretrained multilingual model mBART50[24] (Tang et al., 2020), which already includes Pashto and English as built-in languages. mBART50 is an extension of mBART (Liu et al., 2020) additionally trained on collections of parallel data with a focus on English as source (*one-to-many* system or mBART50 1–to–*n* for short) or target (*many-to-one* system). As of March 2021 the *n*–to–1 system was not available for download; therefore, we used the *many-to-many* (mBART50 *n*–to–*n* for short) version as a replacement. As regards mBART50 1–to–*n*, our preliminary experiments showed that the bare model without further fine-tuning gave in the English→Pashto direction results similar to mBART50 *n*–to–*n*. We also confirmed that mBART50 1–to–*n* gives very bad results on Pashto→English as the system has not been exposed to English during pretraining. Consequently, our experiments focused on mBART50 *n*–to–*n* for both translation directions; being a multilingual model, this will also reduce the number of experiments to consider as the same system is trained at the same time in both directions.

Although these models have already processed English and Pashto texts (not necessarily mutual translations) during pretraining, fine-tuning them on English–Pashto parallel data may improve the results. Therefore, apart from evaluating the plain non-fine-tuned mBART50 *n*–to–*n* system, we *incrementally* fine-tuned it in three consecutive steps:

1. First, we fine-tuned the model with a very small parallel corpus of 1,400 sentences made of the TED Talks and Wikimedia files in the clean parallel data set provided for the WMT 2020 shared task on parallel corpus filtering and allignment for low-resource conditions.[25] Validation-based early stopping was used and training stopped after 20 epochs (this took around 20 minutes on one NVIDIA A100 GPU). This scenario may be considered as a few-shot adaptation of the pretrained model.

2. Then, we further fine-tuned the model obtained in the first step with a much larger parallel corpus of 343,198 sentences made of the complete WMT 2020 clean dataset and the first 220,000 sentences in the corpus resulting from the system submitted by Bytedance to the same shared task (Koehn et al., 2020). Training stopped after 7 epochs (around 2 hours and 20 minutes on one A100 GPU).

3. Finally, we additionally fine-tuned the model previously obtained with a synthetic English–Pashto parallel corpus built by translating 674,839 Pashto sentences[26] into English with the model resulting from the second step. The Pashto→English model in the second step gave a BLEU score of 25.27 with the BBC test set, allowing us to assume that the synthetic English generated has reasonable quality. Note that we carried out a multilingual fine-tuning process and therefore the synthetic corpus is used to fine-tune the system in both directions, which may result in a system that is worse than the initial one in the Pashto→English direction. Training stopped after 7 epochs (around 4 hours on one A100 GPU). Only sentences in the original Pashto monolingual corpus with lengths between 40 and 400 characters were included the synthetic corpus.

[24] https://github.com/pytorch/fairseq/blob/master/examples/multilingual
[25] http://www.statmt.org/wmt20/parallel-corpus-filtering.html
[26] Concatenation of all files available at http://data.statmt.org/news-crawl/ps on March 2021 except for news.2020.Q1.ps.shuffled.deduped.gz.

	BBC test	DW test	FLORES devtest
Google	12.84	10.19	9.16
mBART50	2.47	1.53	7.56
+ small	9.93	7.67	8.24
+ small, large	11.85	10.31	**10.82**
+ small, large, synthetic	**18.55**	**12.54**	8.61

Table 39: BLEU scores of the English→Pashto systems. Each column represents a different test set used to compute the score. The first row contains the results for a commercial general-purpose system. The results for mBART50 correspond, from top to bottom, to a non-fine-tuned mBART50 n–to–n system, and this system incrementally fine-tuned with a small parallel corpus of 1,400 sentences, a larger parallel corpus of 343,198 sentences, and a synthetic corpus of 674,839 sentences obtained from Pashto monolingual text.

	BBC test	DW test	FLORES devtest
Google	0.413	**0.374**	**0.345**
mBART50	0.170	0.147	0.284
+ small	0.351	0.301	0.314
+ small, large	0.389	0.341	0.343
+ small, large, synthetic	**0.463**	**0.374**	0.330

Table 40: chrF2 scores of the English→Pashto systems. See table 39 for details.

Validation-based early stopping was applied with a patience value of 10 epochs. The development set evaluated by the stopping criterion was the in-house validation set made of 1,000 sentences curated by the BBC presented in Section A.13.1.

A.13.3 Indicators of quality

Tables 39 and 40 show BLEU and chrF2 scores, respectively, for the English to Pashto systems with different test sets. The evaluation metrics for the Google MT system are also included for reference purposes. Similarly, tables 41 and 42 show BLEU and chrF2 scores, respectively, for the Pashto to English systems. All the scores were computed with `sacrebleu` (Post, 2018).

The test sets considered are the two in-house parallel sets created by BBC and DW as well as the devtest set provided in the FLORES-101 benchmark.

	BBC test	DW test	FLORES devtest
Google	**35.03**	**24.65**	**21.54**
mBART50	19.42	15.30	14.59
+ small	22.55	17.50	14.77
+ small, large	25.27	19.13	17.71
+ small, large, synthetic	25.38	17.88	17.08

Table 41: BLEU scores of the Pashto→English systems. See table 39 for details.

	BBC test	DW test	FLORES devtest
Google	**0.628**	**0.532**	**0.506**
mBART50	0.456	0.431	0.423
+ small	0.512	0.471	0.420
+ small, large	0.527	0.481	0.451
+ small, large, synthetic	0.535	0.477	0.448

Table 42: chrF2 scores of the Pashto→English systems. See table 39 for details.

	Pashto→English	English→Pashto
Google	83.80	68.50
mBART50 (beam width 1)	**85.15**	83.60
mBART50 (beam width 5)	83.15	**92.30**

Table 43: Average human scores for 20 translations generated by 3 of our models and a commercial general-purpose system.

Regarding the mBART50-based models, for the English→Pashto direction, the scores obtained with the non-fine-tuned models for the FLORES test set are considerably higher than those corresponding to the BBC and DW test sets, which suggests that either they belong to different domains, or they contain very different grammatical or lexical structures, or the FLORES corpus was used to pretrain mBART50. This indicates that fine-tuning could provide a twofold benefit: on the one hand, it may allow the model to focus on our two languages of interest, partially forgetting what it learned for other languages; on the other hand, it may allow the model to perform domain adaptation. In the English→Pashto direction each successive fine-tuning step improves the scores, except when the last model is evaluated against the FLORES devtest set, which makes sense as the development set belongs to the domain of the BBC and DW test sets. Notably, the system resulting from the three-step fine-tuning process improved Google's scores as of April 2021. In the Pashto→English direction, the same trend can be observed, although in this case the best mBART50-based system is noticeably behind the scores of Google's system, yet it still provides scores higher than those for the other translation direction.

Human Evaluation. Four senior editors from BBC Pashto were asked to score translations in a blind exercise from 1 to 100, with 100 indicating top quality. The evaluators were provided with four outputs for both English→Pashto and Pashto→English samples; these outputs were obtained from the mBART50-based models with beam widths of 1 and 5, and from Google Translate. Table 43 demonstrates the average scores by human evaluators for 20 selected sentences. This small sample means that the scores are indicative of the model performance, but together with the BLEU scores gives the user partners confidence in the translation quality. Both mBART50-based models performed very strongly, with outcomes significantly better than Google.

A.14 Burmese↔English

This section describes the resources used and the steps followed to build the English–Burmese NMT systems for both translation directions. A fine-tuning of a pretrained model with back-

Corpus	sentences	en tokens	my tokens
ALT (train + dev)	19 088	435 294	707 604
TALPCo	1 372	10 670	12 032
TICO	3 071	70 587	118 502
TED	63 427	1 042 630	1 509 941
Tatoeba	218	1 427	2 156
Global Voices	2 359	42 692	56 964
total	89 535	1 532 713	2 407 199

Table 44: Parallel English–Burmese corpora used to train the NMT systems. The Burmese token count was calculated after applying Pydaungsu.

translated data was carried out, and knowledge distillation was then applied with the aim of obtaining a smaller and faster system.

A.14.1 Corpora

Tables 44 and 45 show, respectively, the English–Burmese parallel and monolingual corpora used for training. As regards parallel corpora, we used corpora downloaded from the OPUS[27] website, the ALT corpus[28] and the TALPCo corpus.[29] The ALT corpus is part of the Asian Language Treebank Project and consists of 20 000 Burmese–English parallel sentences from news articles. TALPCo is a corpus composed of 1 372 Japanese sentences translated into Korean, Burmese, Indonesian, Malay, Thai, Vietnamese and English; we only used the English–Burmese corpora. We discarded some of the corpora available in OPUS because of the poor quality of their content; other corpora from very different and narrow domains (e.g. software) were also discarded.

Concerning monolingual corpora, only NewsCrawl (Bojar et al., 2018) for English and OSCAR[30] for Burmese were used. NewsCrawl was chosen because it belongs to the news domain, which is our target for this project. OSCAR is a huge multilingual corpus obtained by language classification and filtering of the Common Crawl corpus, removing duplicates. It is the largest and cleanest Burmese corpus we could find.

Finally, development and test corpora, whose statistics are shown in Table 46, were produced by automatically aligning and manually validating BBC news articles. Two additional test corpora were used for evaluation: FLORES 101[31], which is a multilingual translation benchmark dataset for 101 languages, and the ALT test, a part of the ALT corpus mentioned above.

Preprocesing. We used SentencePiece[32] (Kudo and Richardson, 2018) to split words into subwords. For each student model trained (see Section A.14.3), a SentencePiece model was trained

[27] http://opus.nlpl.eu/
[28] https://www2.nict.go.jp/astrec-att/member/mutiyama/ALT/
[29] https://github.com/matbahasa/TALPCo
[30] https://oscar-corpus.com
[31] https://github.com/facebookresearch/flores
[32] https://github.com/google/sentencepiece

Corpus	sentences	Tokens
News Crawl	91 580 474	2 215 550 043
OSCAR	1 192 914	56 074 383

Table 45: Monolingual Burmese and English corpora. The Burmese token count was calculated after applying Pydaungsu.

Corpus	sentences	en tokens	my tokens
development	1 000	16 404	26 327
test	1 000	17 068	27 368
ALT (test)	1 018	22 937	37 297
FLORES	1 012	21 901	34 182

Table 46: Development and test corpora. The Burmese token count was calculated after applying Pydaungsu.

on the training corpora with a vocabulary size of 10 000 tokens. For fine-tuning the mBART50 pretrained model we used its own SentencePiece model.

SentencePiece does not need the corpus to be tokenized in advance. In spite of this, we used Pydaungsu [33] to segment the Burmese corpora into words and harmonize the word tokenization in the different training corpora. This was needed because the introduction of blank spaces between words is not mandatory in Burmese and if they are introduced it is not regularly. After tokenization with SentencePiece, sentences with more than 100 tokens were removed. Table 47 reports the size of the corpora after this preprocessing with the SentencePiece model of the pretrained mBART50 NMT system.

A.14.2 Language resources

Our systems are based on the pretrained multilingual model mBART50 (Tang et al., 2020).[34]. This system was trained with the ALT corpus and consequently its SentencePiece model processes better texts segmented in a similar way. After analysing some segmenters, we found that Pyidaungsu segments the text similarly to the ALT corpus. Thus, we used Pyidaungsu for preprocessing Burmese corpora for the mBART50 fine-tuning and student training.

[33] https://github.com/kaunghtetsan275/pyidaungsu
[34] https://github.com/pytorch/fairseq/blob/master/examples/multilingual

Corpus	Languages	sentences	SL tokens	TL tokens
parallel	English–Burmese	87 435	2 217 760	4 614 047
NewsCrawl	English	4 731 302	152 009 028	-
OSCAR	Burmese	731 421	44 877 903	-

Table 47: Size of the corpora used to build the NMT systems after preprocesing. For the English NewsCrawl corpus, only the size of the subset that has been used for training is displayed. Token counts were calculated after splitting with SentencePiece.

In order to properly evaluate MT systems which translate to Burmese, it is necessary to have the reference translations and the output of the system segmented in the same way. We achieve this by using the segmenter used for Burmese in the WAT2020 [35] translation task. This segmenter splits sentences into characters with their respective diacritics.

A.14.3 Model architecture and training

The NMT models were obtained by fine-tuning the largest downloadable mBART50 pretrained model with the parallel data described in Table 47 and performing iterative back-translation with the monolingual corpora described in the same table. With the best fine-tuned mBART50 model we then carried out sequence-level knowledge distillation (Kim and Rush, 2016) and multi-task data augmentation (Sánchez-Cartagena et al., 2021) to train a student model. In both cases, the fairseq toolkit (Ott et al., 2019) was used.

Fine-tuning mBART50 We used the n-1 mBART50 model for the Burmese–English translation direction and the 1-n mBART50 model for the English–Burmese translation direction. Validation-based early stopping was applied with a patience value of 10 epochs. For this we used as development set the in-house development set made of 1 000 parallel sentences curated by the BBC. We selected the checkpoint that obtained the highest BLEU (Papineni et al., 2002) score on the development set. For Burmese, we preprocess the input with the Pyidaungsu segmenter.

First, we fine-tuned the pretrained models with a parallel data described in Table 47 and then we tried to improve them by making use of the monolingual corpora through back-translation. We took advantage of the fact that we are building systems for both directions and applied an iterative back-translation algorithm that simultaneously leverages monolingual Burmese and monolingual English data. The process can be outlined as follows:

1. Fine-tune mBART50 systems using English—Burmese parallel corpora.
2. Back-translate the monolingual data.
3. Fine-tune the systems with parallel corpora and synthetic corpora generated at step 2 and go back to step number 2.

Steps 2-3 were executed twice. After fine-tuning, we translated monolingual corpora with the best systems to generate the training sets for the student models.

Knowledge Distillation and Data Augmentation Just as with mBART50, we trained a student model for each translation direction. All student models were trained using the Transformer base architecture. Early stopping was based on perplexity on the development set and patience was set to 6 validations, with a validation carried out every 5 000 updates. Batch size was set to 4 000 tokens. No hyperparameter tuning was performed.

Before training the student models, we used the parallel corpora described in Table 44 to train a Bicleaner (Sánchez-Cartagena et al., 2018) model. Bicleaner was used in order to filter the

[35] http://lotus.kuee.kyoto-u.ac.jp/WAT/WAT2020/

synthetic corpora generated by the mBART50 models. Different Bicleaner scores were tested and a score of 0.7 was finally selected.

Since mBART50 was trained only with the ALT corpus for the Burmese language, which is a small corpus, we could not perform knowledge distillation as proposed in the paper mentioned above (Kim and Rush, 2016). Instead, we used monolingual source and target language corpora, which we translated with the models obtained in the previous step, in addition to the original parallel corpus. We tried the following combinations:

1. Parallel corpora and synthetic corpora at source (back-translation).
2. Parallel corpora and synthetic corpora at target (forward-translation).
3. All corpora available.

The best result for Burmese–English was obtained using all the corpora; for English–Burmese the best result was obtained using back-translation.

With the Aim of improving the student models to be delivered, we applied a multi-task learning approach for data augmentation (Sánchez-Cartagena et al., 2021). This approach consists of generating new synthetic sentence pairs by applying simple transformation to sentence pairs in the training corpus and using tags to mark each transformation, as in multilingual NMT. Specifically, we applied Reverse and Replace tasks on all training corpora (parallel + synthetic). After training, we fine-tuned models with parallel corpora only.

With this method, the English–Burmese model improved significantly, while the Burmese–English model obtained results close obtained when training without data augmentation.

A.14.4 Indicators of quality

Tables 48 and 49 show the BLEU (Papineni et al., 2002) and chrF++ (Popović, 2017) scores computed on the test sets, for the different steps in the development of the MT systems. As a reference, we also show the scores obtained by the translation obtained with Google Translate[36] on 29th Nov 2021.

Google scores for Burmese–English were obtained by removing all spaces from the Burmese side of the test sets before translating; keeping the original segmentation or applying the Pyidaungsu segmenter resulted in worse results. We noticed that Google Translate systems improved since we started building our systems, which suggests that their systems were trained on data that was not available at that time.

The results show that knowledge distillation allow us to obtain smaller and faster student models with scores similar to those obtained by the fine-tuned mBART50 models. It is remarkable the positive impact of the data augmentation on the English–Burmese direction. Depending on the test corpus, our results are at the same level of Google or even better.

[36] https://translate.google.com/

Model	Test	BLEU	chrF++
mBART50	BBC	17.70	0.38
	ALT	**36.40**	0.54
	FLORES	28.25	0.49
mBART50 finetuned	BBC	**21.70**	0.39
	ALT	32.30	0.50
	FLORES	24.70	0.43
Student	BBC	19.07	0.37
	ALT	25.62	0.43
	FLORES	20.30	0.39
Student with MTL	BBC	21.47	0.40
	ALT	30.02	0.48
	FLORES	23.90	0.43
Google Translate	BBC	21.12	0.39
	ALT	25.95	0.44
	FLORES	**32.56**	0.50

Table 48: Automatic evaluation results for English→Burmese obtained for the different development steps of the MT systems.

Model	Test	BLEU	chrF++
mBART50	BBC	15.45	0.45
	ALT	29,73	0.56
	FLORES	22.32	0.50
mBART50 finetuned	BBC	16.98	0.47
	ALT	**30.80**	0.57
	FLORES	23.60	0.51
Student	BBC	16.13	0.46
	ALT	29.04	0.56
	FLORES	20.83	0.50
Student with MTL	BBC	15.80	0.46
	ALT	28.08	0.56
	FLORES	20.29	0.49
Google Translate	BBC	**24.13**	0.54
	ALT	26.19	0.53
	FLORES	**25.28**	0.53

Table 49: Automatic evaluation results for Burmese→English obtained for the different development steps of the MT systems.

A.15 Yoruba↔English

In this section we describe models trained for English–Yoruba and Yoruba–English translation. Since Yoruba is a very low resource language, we resorted to fine-tuning of the multilingual M2M100 model, which was trained also on Yoruba data (Fan et al., 2021).

A.15.1 Corpora

The parallel corpora available for Yoruba–English translation is summarized in Table 50. Apart from OPUS (Tiedemann and Thottingal, 2020), there is one other publicly available dataset, called MENYO-20k (Adelani et al., 2021).

There is also a small amount of monolingual data in Yoruba, namely in the CC-100 corpus (Conneau et al., 2020; Wenzek et al., 2020). For our English monolingual corpus, we used a part of the News Crawl of 2020, which contains over 1M sentences.

Corpus		sentences	En tokens	Yo tokens
OPUS	CCAligned	175 193	2 666 712	3 044 766
	GlobalVoices	136	1 921	2 243
	GNOME	10 234	51 761	58 548
	Tatoeba	37	880	153
	Ubuntu	141	4 470	735
	wikimedia	8 521	181 705	216 730
	XLEnt	51 173	143 511	144 218
MENYO-20k		10 070	1 050 132	1 440 705
Total		255 505	3 220 398	3 666 882

Table 50: Size of the parallel corpora used to build the English–Yoruba NMT systems.

Corpus	sentences	tokens
CC-100 (yo)	76 533	864 424
News Crawl 2020, part 1 (en)	1 276 848	24 844 116

Table 51: Size of the monolingual corpora used to build the English–Yoruba NMT systems.

A.15.2 Model architecture and training

We begin with the HuggingFace implementation of the M2M100 model[37]. We train an initial translation model by fine-tuning the pre-trained model with the parallel data.

We prepare synthetic data by backtranslating both the monolingual and parallel data with the initial models, and we fine-tune the M2M100 model again on the extended training dataset.

[37] https://huggingface.co/docs/transformers/model_doc/m2m_100

A.15.3 Indicators of quality

Table 52 shows SacreBLEU scores computed on the development set and on the BBC test set. The results in row "M2M + fine-tuning" were obtained by fine-tuning the M2M100 model on the (very small amount of) parallel data only. Row "M2M + fine-tuning + BT" shows scores obtained with backtranslation. We also show the BLEU scores achieved Google Translate. We see that in the English–Yoruba direction, both our models and Google Translate score poorly. An interesting aspect of this translation direction is the negative correlation between the scores on the development set (a part of the MENYO-20k corpus) and on the BBC test set. We hypothesize that the cause of this effect is the domain mismatch between the two data sets. For Yoruba–English, the results are better, which is probably due to larger amount of English monolingual data available for backtranslation.

Model	English–Yoruba		Yoruba–English	
	dev	test	dev	test
M2M + fine-tuning	10.3	4.0	16.2	10.5
M2M + fine-tuning + BT	9.8	4.5	18.4	12.0
Google Translate	5.1	7.4	20.4	17.1

Table 52: SacreBLEU results for English↔Yoruba

A.16 Urdu↔English

This subsection describes the resources, training methodology, and evaluation experiments for the Urdu-English and English-Urdu NMT systems. The training procedure primarily involved fine-tuning the mBART50 model (Tang et al., 2020) with parallel corpora, followed by distilling the knowledge of this teacher model into a smaller student model.

A.16.1 Corpora

A total of 23 parallel corpora were used to train the Urdu systems, as shown in Table 53. Most of the corpora were obtained from the OPUS project (Tiedemann, 2012). In addition, these were supplemented by parallel corpora from other works. This includes the Bible and Quran corpora (Jawaid and Zeman, 2011), the Emille corpus (McEnery et al., 2000), the Indian Parallel Corpora (Post et al., 2012), the PMIndia corpus (Haddow and Kirefu, 2020), the Penn Treebank corpus (Marcus et al., 1993), the TDIL corpus (Jha, 2010), the CVIT Press Information Bureau (Philip et al., 2021) and the Mann Ki Baat (Siripragada et al., 2020) corpora respectively.

The monolingual corpora used are described in Table 54. For Urdu, given the absence of News Crawl corpus for this language (or, to the best of our knowledge, any news domain corpus), the Charles University Urdu Monolingual corpus (Jawaid et al., 2014) was chosen. This corpus covers various domains, including the news domain. In addition, the Urdu Wikipedia dump[38] and the Urdu

[38] https://dumps.wikimedia.org/urwiki/latest/

side of the CCMatrix parallel corpus were obtained and concatenated with the Charles University corpus for larger corpus size. For English, the News Crawl 2020 dataset[39] was chosen.

Corpus	sentences	en tokens	ur tokens
Bible	6 423	135 609	161 417
CCAligned v1	581 672	6 030 644	7 405 803
Emille	9 773	116 702	174 599
GlobalVoices v2018q4	1 617	27 576	32 894
GNOME v1	3 088	13 250	16 165
Indian Parallel Corpora	7 428	104 713	118 294
JW300	18 708	257 776	217 355
CVIT MKB v0.0	977	17 284	19 593
Mozilla-I10n v1	15 441	79 822	108 410
OpenSubtitles v2018	14 909	111 868	136 213
CVIT PIB v1.3	80 256	1 754 799	2 270 043
PMIndia v1	3 300	59 441	76 095
QED v2.0a	18	138	155
Quran	4 752	106 472	118 150
Tanzil v1	607 103	12 254 803	14 628 252
Tatoeba v2021-07-22	1 531	9 803	11 551
TDIL-DC	1 021	18 986	25 949
TED2020 v1	13 851	222 606	269 970
tico-19 v2020-10-28	2 403	48 075	61 956
Treebank-3	1 326	25 000	35 963
Ubuntu v14.10	992	3 887	5 188
wikimedia v20210402	11 209	201 422	246 016
XLEnt	22 705	49 714	51 425
total	1 410 503	21 650 390	195 102 072

Table 53: Parallel corpora used to train the Urdu-English systems

Preprocessing Given the wide variety of sources used in both parallel and monolingual corpora, thorough preprocessing and curation were required to maintain the quality of the training corpora. Thus, various corpus and language-specific cleaning techniques built on standard preprocessing methodologies were developed and used. Developing such customized methodologies was feasible given the author's native fluency in Hindi, a language phonetically identical to Urdu due to their shared history as the unified Hindustani language[40]. However, since Hindi and Urdu use different scripts (making comprehension of written text infeasible), Google Text-to-Speech was used to convert text to audio. This could then be used to identify the quality of monolingual Urdu sentences as well as Urdu-English bitext, based on Urdu's aforementioned phonetic similarities to Hindi.

[39] https://data.statmt.org/news-crawl/en/
[40] https://en.wikipedia.org/wiki/Hindustani_language

Corpus	sentences	tokens
Charles University corpus (Ur)	5 027 008	76 625 011
Wikipedia (Ur)	1 320 239	18 542 594
CCMatrix mono (Ur)	5 433 298	70 766 513
News Crawl (En)	16 279 967	366 345 750

Table 54: Monolingual Urdu and English corpora used for Back Translation in the Urdu-English systems

Some parallel corpora - particularly CCAligned, JW300, OpenSubtitles, QED, and XLEnt - provide confidence scores for the reference translations. The first stage of filtering for such corpora, therefore, involved using threshold values to filter out poor-quality translations. These threshold values, shown in Table 55, were arrived at by manually checking the quality of the filtered data (as described previously) on using threshold scores at steps of 0.1 and then 0.01. Similarly, the Indian Parallel Corpora (Post et al., 2012), created using crowdsourced translators from Amazon Mechanical Turk, provides results from a separate task in which Turks were asked to vote for the best translation among the ones provided by the translators. These votes provided another opportunity for filtering - for this corpus, only translations with two or more votes were chosen for inclusion in the training corpus. It is worth noting here that another parallel corpus, CCMatrix (Schwenk et al., 2021), that had a filtered size of 5.4M sentences, was initially considered for inclusion in the training dataset. Threshold-based filtering was attempted on this corpus, but it could not work as the provided confidence scores for the reference translations were empirically found to be highly noisy and irregular, as was the corpus itself. This is proven by results in subsection A.16.3, where inclusion of CCMatrix in the training set is shown to hurt BLEU by as high as 5-7 BLEU points. It was thus excluded from the training corpora.

Threshold-based filtering was followed by length-based filtering, where sentences that were too short or too long were discarded. In addition, parallel sentences were also filtered based on length ratios or length differences. Filtering based on length ratio was used on short/medium-length sentences (defined as sentences with lengths less than or equal to 25 words), while long sentences (defined as sentences with lengths greater than 25 words) were filtered using length differences. The former was empirically observed to be able to discard bitext with poor reference translations, while the latter was able to filter out sentences where one side (typically Urdu) was a paraphrasing of the other (typically English) and contained considerably lesser information - despite having reasonable length ratios. Since these corpora were crawled from documents on the Web, there were quite a lot of such paraphrased sentences that could hurt translation quality if used to train NMT models. Next, language identification-based filtering was attempted to potentially filter out noisy sentences from monolingual and parallel corpora that belonged to another language. But in practice, both Facebook's FastText and Google's CLD3, were found to misclassify Urdu sentences as Arabic or Persian, so this approach had to be discarded. Instead, a script-based filtering mechanism was applied where sentences with non-Urdu scripts were filtered out from Urdu corpora and non-Latin scripts filtered out from English corpora, using the appropriate unicode characters to identify each script. Finally, sentences with more special characters or digits than words were also filtered out. The cleaned corpus was then tokenized using SentencePiece (Kudo and Richardson, 2018) and used for training the models, as described ahead.

Corpus	CCAligned	JW300	OpenSubtitles	QED	XLEnt
Threshold	1.05	0.37	0.6	0.65	2

Table 55: Threshold values used to filter parallel corpora that provided confidence scores

A.16.2 Model architecture and training

In light of Facebook's multilingual NMT systems recently succeeding at WMT21 (Tran et al., 2021), there has been an increased interest in adapting pre-trained multilingual models for downstream translation tasks. The inclusion of Urdu in the pretraining and the fine-tuning corpora of one of the SOTA multilingual NMT systems, mBART50, suggested the utility of leveraging this pre-trained model for training our Urdu systems. The first stage of the training procedure, therefore, involved fine-tuning of the 1-n and n-1 checkpoints[41] of the mBART50 model using Ur-En and En-Ur parallel corpora (Table 54) respectively. A baseline trained additionally on CCMatrix was also considered, as mentioned in Section A.16.1. Post this, two rounds of iterative Back Translation (BT) using the English and the Urdu monolingual corpora (Table 54) were carried out to improve the performance further. For Urdu, the first iteration of back-translation used the Charles university corpus and the Urdu Wikipedia dump, while the second iteration also used the Urdu portion of the noisy CCMatrix parallel corpus.

The second stage of the training pipeline focused on improving efficiency. Towards this end goal, knowledge distillation from the developed teacher model (fine-tuned mBART) to a faster and lighter bilingual student model was carried out. This was done by using the trained teacher models to generate translations of all available Urdu and English data in our corpora. The student model was then trained to mimic the teacher model using the original corpus as the source and the generated translations as the target. After this, various optimization techniques were experimented with to improve further the efficiency of the student model, including shortlisting (Schwenk et al., 2007) and quantization (Behnke et al., 2021). While shortlisting significantly damaged the model's performance, quantization using intgemm8 and intgemm16 was observed to improve efficiency by 2.5x, with negligible reduction in performance, and so was incorporated into the final model.

Experimental settings: Given that the mBART checkpoints were only available on Fairseq (Ott et al., 2019), fine-tuning these to develop the teacher models was carried out on Fairseq as well. However, the student models were trained on Marian (Junczys-Dowmunt et al., 2018) due to its greater resource and computational efficiency. Following the mBART checkpoints, the teacher models used the mBART-large architecture (12 encoders and 12 decoders). Label smoothed cross-entropy loss (with label smoothing=0.2) was used as the loss criterion. An inverse sqrt learning rate scheduler initialized with a learning rate of 3e-05 was adopted. A dropout of 0.3 was used for regularization. Optimization was done using the Adam optimizer, with an epsilon value of 1e-06 and beta values of (0.9, 0.98). Validation was carried out every 10000 updates, with the patience value set to 10 validations for early stopping.

For the student model, all possible architecture combinations of 10, 8, and 6 encoders; and 6 and 4 decoders were experimented with for improving efficiency. The model of size 8 encoders and 4 decoders was observed to perform comparably to the largest model (10 encoders and 6 decoders) while also maximising efficiency - and was hence chosen as the architecture size of the final student

[41] https://huggingface.co/facebook/mbart-large-50-one-to-many-mmt

Model	Ur-En	En-Ur
mBART50 (original)	6.9	15.1
mBART50 ft. 1.5M	31.9	32.1
mBART50 ft. 1.5M + 5.5M (CCMatrix)	24.9	27.3
mBART ft. 1.5M + BT1	33.5	32.1
mBART ft. 1.5M + BT2	**33.6**	**32.6**

Table 56: BLEU scores of Ur-En and En-Ur teacher baseline models on our 40K Urdu News test set

model. Beam size was also reduced from 5 to 1 for the same reasons, which resulted in a slight decline in performance in Ur-En, but an improvement for En-Ur. An inverse sqrt learning rate scheduler initialized with a learning rate of 3e-04 was used to help converge the model. No dropout was used. Optimization was done using the Adam optimizer, with an epsilon value of 1e-09 and beta values of (0.9, 0.98). Validation was done every 3000 updates, with the patience value set to 10 validations for early stopping.

A.16.3 Indicators of quality

For Urdu, given the absence of standardized test datasets in WMT or other works, a test dataset of 40K sentences constructed from the PMIndia, PIB and GlobalVoices corpora was used for evaluation, in addition to the BBC test set. Table 56 shows the BLEU (Papineni et al., 2002) scores of various baselines of teacher models evaluated on the 40K test set. It is worth noting that inclusion of CCMatrix in the training corpus results in a significant decline in performance, by 7 and 5 BLEU points in Ur-En and En-Ur directions, respectively. Table 57 shows the corresponding results on the BBC test set, along with the translation time per sentence on 1 CPU core. For all baselines, efficiency was evaluated on 1 Peta4-Skylake node on the Cambridge CSD3 cluster, with 5980 MB of RAM.

Fine-tuning mBART significantly improves the performance, as expected. Moreover, knowledge distillation does not majorly hurt BLEU, but does drastically improve efficiency. In contrast to the Fairseq models, including mBART50 and the final Teacher models, that were found to crash with an Out of Memory (OOM) error on our CPU nodes, the developed student models in Marian were found to be able to translate one sentence in 0.49s in both directions. This figure reduced to 0.2s and 0.186s respectively, when quantization using intgemm16 was used. This resulted in a marginal decline in BLEU in the Ur-En and an improvement in the En-Ur direction. On using all 8 CPU cores, the translation time per sentence further reduced to 0.036s and 0.04s respectively, about 2.25x faster than Google Translate. However, Google Translate does outperform our system in terms of BLEU scores.

A.17 Turkish↔English

Our strategy for English-Turkish is to explore advances in transfer learning, in particular, multilingual pre-training. Concretely, we fine-tune a large pre-trained multilingual BART model (Liu et al., 2020), namely mBART25, on an English-Turkish parallel corpus. We fine-tune the architecture in each direction independently, which yielded better results than joint learning both directions in our investigation. We then experiment with combining parallel and synthetic data obtained via

Model	Ur-En	En-Ur	Time* (Ur-En/En-Ur)
Google Translate	**49.5**	**44**	0.09s/0.09s
mBART50 (original)	29.7	13.5	Crashes on 1 CPU core
Teacher model (beam size=5)	42.8	34.3	Crashes on 1 CPU core
Teacher model (beam size=1)	41.1	33.5	- (crashes on CPU)
Student (best, beam size=5)	41.8	35.2	0.49s/0.49s
Student (shortlist, beam size 5)	32	23.7	0.28s/0.249s
Student (intgemm16, beam size 5)	**41.8**	35.2	0.39s/0.39s
Student (intgemm16, beam size 1)	41.3	**35.3**	0.2s/0.267s
Final	41.3	35.3	0.2s/0.18s
Final (8 CPU cores)	41.3	35.3	0.036s/0.04s

Table 57: BLEU scores and translation times on the BBC test sets. *Time indicates translation time per sentence on 1 CPU core, unless otherwise indicated, for the Ur-En and En-Ur directions respectively. For Google Translate, the time to receive a translation response is shown.

back-translation. We incorporate back-translated data using a language-token, which we use to identify synthetic inputs. This is similar to using language-tokens to identify domains (Chen et al., 2019) or pragmatic features such as politeness (Sennrich et al., 2016b). Concretely, we initialise a 'synthetic-language' token with the embedding of the source-language token, from a pre-trained multilingual BART model (Liu et al., 2020), mBART25 in particular, and fine-tune the entire architecture on a combination of parallel and synthetic corpora. We find that both synthetic data and this specific way of incorporating it contribute to our best results.

A.17.1 Corpora

The parallel corpora we use is from OPUS (Tiedemann and Thottingal, 2020). We selected corpora which were closer to the news domain (i.e., Bianet, ELRC2922, GlobalVoices, GoURMET, SETIMES, TildeMODEL, infopankki). Table 58 lists the resources.

Corpus	Sentences	Tokens (en)	Tokens (tr)
Bianet	35 080	740 305	582 413
ELRC2922	2 367	48 873	36 846
GlobalVoices	7 592	140 935	104 489
GoURMET	1 308 303	43 465 382	37 556 607
infopankki	44 635	511 544	394 790
SETIMES	207 678	4 428 278	3 654 669
TildeMODEL	1 584	39 513	34 406
Total	1 607 239	49 374 830	42 364 220

Table 58: Parallel corpora used to train the Turkish-English systems

For this language pair, we use publicly available back-translations. We start with back-translations

made available by the Tatoeba Translation Challenge (Tiedemann, 2020),[42] and, based on our domain of interest, we use `wikinews`. In addition, we reuse back-translations made available by the University of Edinburgh—they back-translated monolingual NewsCrawl 2016 and 2017 datasets for their WMT18 submission (Haddow et al., 2018). We use a subsample of 1.5 million sentences.

For word-segmentation and tokenisation, we rely on BPE-segmentation (Sennrich et al., 2016d) as implemented in `sentencepiece`.[43]. We use mBART25's own sentencepiece model (Liu et al., 2020). Note that this is a joint vocabulary for all supported languages.

For evaluation in development phase, we use the WMT newstest2016 (3000 sentence pairs) as a development set (i.e., used for early stopping) and WMT newstest2017 as a devtest set (i.e., model selection).

A.17.2 Model architecture and training

A.17.3 Indicators of quality

Table 59 reports translation quality in terms of SacreBLEU (Post, 2018).[44] As expected, backtranslation contributes towards the best results. The synthetic-language token simplifies a hyperparameter search (otherwise the proportion of gold vs. synthetic data would have to be determined by trial and error) and improves the system further. Table 60 compares the performance of the initial system (i.e., mBART fine-tuned on parallel resources) to that of the final system (i.e., fine-tuned towards a combination of parallel and synthetic data using a synthetic-language token) on the small BBC dev set.

Model	English-Turkish		Turkish-English	
	dev	devtest	dev	devtest
Parallel	20.6	22.2	26.5	26.8
+ Tatoeba wikinews	21.2	22.2	28.7	28.1
+ UEDIN NewsCrawl	20.8	23.4	29.0	28.6
+ synthetic-language token	22.0	24.3	28.6	28.7

Table 59: SacreBLEU results on English-Turkish WMT dev and devtest sets.

Model	English-Turkish	Turkish-English
Initial	20.6	32.0
Final	21.3	33.7

Table 60: SacreBLEU results on English-Turkish BBC dev.

[42] https://github.com/Helsinki-NLP/Tatoeba-Challenge/blob/master/data/Backtranslations.md
[43] https://github.com/google/sentencepiece
[44] Signature: `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0`.

ENDPAGE

GoURMET

H2020-ICT-2018-2 825299

D5.5 GoURMET Final progress report on integration