UTTER

Unified Transcription and Translation for Extended Reality (UTTER)

Horizon Europe Research and Innovation Action Number: 101070631

D6.1 – First report on efficient inference

Nature	Report	**Work Package**	WP6
Due Date	28/03/2024	**Submission Date**	28/03/2024
Main authors	Alexandra Birch (UEDIN)		
Co-authors	José Souza (UNB), Amin Farajian (UNB), Barry Haddow (UEDIN), Tsz Kin Lam (UEDIN), Pinzhen Chen (UEDIN)		
Reviewers	Marcely Zanon Boito		
Keywords	Efficient, inference, compact, evaluation, tools		
Version Control			
v0.1	**Status**	Draft	14/03/2024
v1.0	**Status**	Final	28/03/2024

This project has received funding from the European Union's Horizon Europe Research and Innovation programme under Grant Agreement No 101070631 and from the UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee (Grant No 10039436).

Contents

List of Figures

Abstract

This deliverable reports on the work done in WP6 on efficient inference, model compression and tools evaluation, usage, sharing and adaptation of the models. For the first half of this project we completed an extensive survey work on efficient natural language processing. This led us to other more focussed work on model distillation, lexical shortlisting and multimodal model compression with discreet speech units. We have also completed extensive work on an evaluation framework described in the toolkit called TowerEval.

1 Contributors

Task	Who is reporting	Paper
T6.1	Birch §3.1	Marcos Treviso, Ji-Ung Lee, Tianchu Ji, Betty van Aken, Qingqing Cao, Manuel R. Ciosici, Michael Hassid, Kenneth Heafield, Sara Hooker, Colin Raffel, Pedro H. Martins, André F. T. Martins, Jessica Zosa Forde, Peter Milder, Edwin Simpson, Noam Slonim, Jesse Dodge, Emma Strubell, Niranjan Balasubramanian, Leon Derczynski, Iryna Gurevych, Roy Schwartz "Efficient Methods for Natural Language Processing: A Survey" MIT Press 2023
T6.1	Haddow §3.2	Nikolay Bogoychev, Pinzhen Chen, Barry Haddow, Alexandra Birch "Large language model inference with lexical shortlisting", DAI-Workshop AAAI 2024.
T6.1	Birch §3.3	Guillem Ramírez, Matthias Lindemann, Alexandra Birch, Ivan Titov "Distil: Optimising API Calls to Large Language Models", (Under Review) ARR
T6.1	Chen §3.4	Zhanghao Hu, Yijun Yang, Junjie Xu, Yifu Qiu, Pinzhen Chen. EEE-QA: Exploring effective and efficient question-answer representations. Accepted to LREC-COLING 2024.
T6.2	Lam §4.1	Tsz Kin Lam, Alexandra Birch, Barry Haddow "Compact Speech Translation Models via Discrete Speech Units Pretraining", (Under Review) ARR
T6.3	de Souza §5.1	Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, André F. T. Martins. "Tower: An open multilingual large language model for translation-related tasks", 2024 arXiv

Table 1: List of publications to be discussed

2 Introduction

2.1 Objectives

Proposal

This WP addresses the call text "proposed solutions should be energy efficient" by optimising model inference. Efficient inference is critical to applications. In our customer support and online meeting assistant scenarios, latency is critical to the user experience. Beyond the immediate proposal, we anticipate that third parties will benefit from reduced computational requirements to build models on custom data or integrate inference into their applications. A second order benefit of efficiency is improved privacy because it supports application developers running inference locally rather than on the cloud. WP6 has the following main goals: optimise training speed and scalability; reduce inference latency and throughput; decrease the size of the model. Many methods trade between efficiency and quality of the model. There is no single speed cutoff nor a single acceptable level of quality loss, but rather a range of options depending on application requirements. For each of these goals, our aim is to push the Pareto frontier i.e. with faster speed but the same quality loss or better quality for the same speed.

Work Completed

In the first half of the project we focussed on inference speed. We made an initial contribution to the hard problem of lexical shortlisting for large language models. This is pioneering work with potentially many avenues for follow-up which we will consider in the second half of the project. We also looked at more efficient use of large language models (LLMs) by caching and distilling the responses from a LLM. Finally we explored effetive and efficient question-answer representations. In terms of task two on model compression, we also started work on smaller models for multimodal speech translation using discrete speech units. Finally for the third task on tools for evaluation, usage and sharing, we delivered TowerEval which is an evaluation framework for large language models.

All this work was guided by the insights and knowledge which were distilled in our survey paper on efficiency which is the first piece of work reported in this deliverable.

3 Task 6.1: Inference Speed (UEDIN)

Proposal

Inference speed is achieved by both specific model architecture and code changes. We will use teacher-student knowledge distillation to train high quality teacher models and then distill them into smaller, less computationally demanding student models. The student models will then be further sped up by using device specific low precision decoding architecture. For CPU inference, we will use 8-bit integer quantisation to get the most out the available x86 SIMD instructions. For GPU inference, we will use fp16 operations, and we will explore the possibility to use 8-bit integer quantisation. Orthogonal to above, we will try reduce output layer size by excluding unlikely tokens from it. This reduces computational cost of the single largest matrix multiplication in the

model and contributes significantly to model decoding. There are two common ways for this: through a lexical shortlist generated by an IBM model or through KNN clustering at output layer.

Proposal highlights

- Teacher-student knowledge distillation
- Quantisation of models
- Lexical shortlisting to reduce output vocabulary

Summary of completed work

In the task we have completed an extensive survey of all efficiency work relating to natural language processing (NLP). This survey gave us insight to start work on lexical shortlisting which although extremely effective for machine translation, is harder for LLMs with more ambiguous output spaces. We then investigated a cache and distill approach to efficiently calling large language models, where more expensive model output is used to train smaller local models. We finally present work on efficiently performing multiple choice question answering.

3.1 Survey on Efficient Methods for Natural Language Processing

Recent work in NLP has yielded appealing results from scaling model parameters and training data; however, using only scale to improve performance means that resource consumption also grows. Such resources include data, time, storage, or energy, all of which are naturally limited and unevenly distributed. This motivates research into efficient methods that require fewer resources to achieve similar results. This survey synthesizes and relates current methods and findings in efficient NLP. They provide both guidance for conducting NLP under limited resources, and point towards promising research directions for developing more efficient methods.

We address this work to two groups of readers: (1) Researchers from all fields of NLP working with limited resources; and (2) Researchers interested in improving the state of the art of efficient methods in NLP. Each section concludes with a discussion of limitations, open challenges, and possible future directions of the presented methods. We start by discussing methods to increase data efficiency, and continue with methods related to model design. We then consider efficient methods for the two typical training setups in modern NLP: pre-training and fine-tuning. We then discuss methods for making inference more efficient. While we mainly focus on algorithmic approaches, we provide appropriate pointers regarding hardware that are connected to the scale at which we expect to deploy a model. We then discuss how to quantify efficiency and what factors to consider during evaluation, and, finally, how to efficiently decide upon the best suited model.

To guide the reader, Figure 1 presents a typology of efficient NLP methods considered in this survey. This work is reported in Treviso et al. (2023).

3.2 Large Language Model Inference with Lexical Shortlisting

In machine translation, the idea of *lexical shortlisting*, where the vocabulary size is reduced based on the input sentence, can significantly speed up inference, especially on CPU. Lexical shortlisting

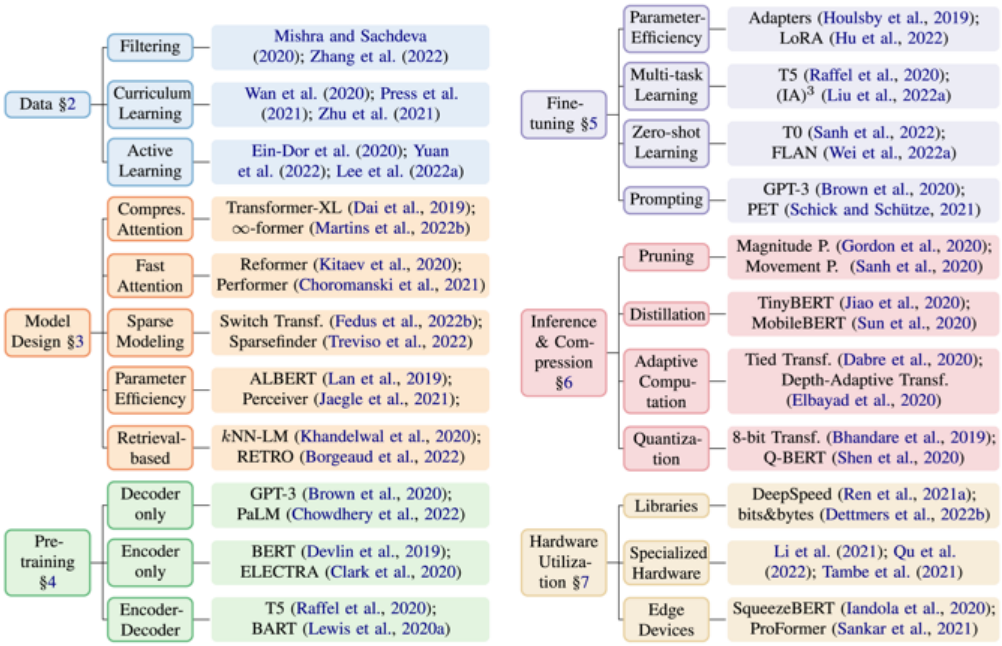

Figure 1: Typology of efficiency studies

works by considering a sub-vocabulary, and only loading the parts of the embedding matrix that correspond to the sub-vocabulary. The selection of the sub-vocabulary is performed by exploiting a word-translation table computed from aligning the training corpus. In this work we investiagte whether the technique can be applied to inference in LLMs.

For LLMs, we have to use a different strategy to select the sub-vocabulary. We propose and experiment with two strategies: script-based token filtering where vocabulary items are removed if they do not belong to the output language, and corpus-based pre-selection where we keep items based on vocabulary hits from a large representative corpus. Our contributions can be summarised as:

- We propose two vocabulary selection methods: one based on writing script filtering, and one based on corpus selection.
- We experiment with models of different families and sizes (LLaMA and BLOOM), and report varying output layer behaviours in terms of performance.
- We perform an extensive controlled study, measuring potential speed-ups for different LLMs and hardware.
- We discuss practical limitations for applying these methods in the wild.

More details of this work are available in our paper (Bogoychev et al., 2024).

3.3 Cache and Distil: Optimising API Calls to Large Language Models

LLMs offer unique capabilities in understanding and generating human-like text. They have gained widespread use in a wide range of applications, such as assistive tools and entertainment bots. However, large models are often very challenging for all but a few companies and institutions to

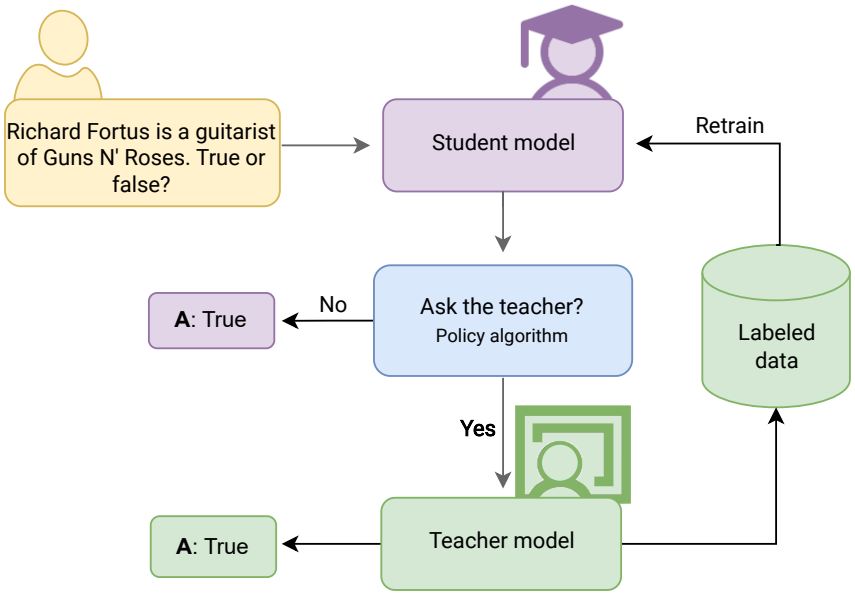

Figure 2: Neural caching (one iteration): A student generates a response to a user request. The policy algorithm determines whether to rely on the student's response or to call an LLM. LLM responses are stored and used to re-train the student as more data becomes available.

run on their infrastructure (Schwartz et al., 2020). Meanwhile, smaller models typically underperform in these applications, at least without additional fine-tuning on task-specific labelled data. Consequently, many applications access LLMs via commercial APIs despite the costs involved and the exposure of their entire request stream to the API providers.

To minimise the costs and data exposure associated with calling the API, we propose to train a smaller language model, which we refer to as *student*, on the LLM's predictions and, as the student gets more accurate, it handles an increasing number of requests. The knowledge of the LLM gets continuously distilled into the smaller model. We refer to this scenario as *neural caching* (see Figure 2), as the student can be thought of as a smart cache. Note though that the student not only remembers what the LLM predicted but also generalises beyond these examples. The goal of this paper is to formalise the neural caching problem and investigate simple ways of approaching it.

The key element in the neural caching scenario is the policy determining which requests the student processes independently. A good policy should weigh the expected immediate user benefit (i.e., if the LLM is substantially more likely to make a correct prediction than the student) and the anticipated benefit for the student (i.e., whether the LLM's prediction will aid in training the student).

The latter underscores its relationship with Active Learning (AL, Settles, 2009; Zhan et al., 2022), although AL is typically associated with soliciting human annotations. In particular, there is a similarity to online AL (Cacciarelli and Kulahci, 2023), where new unlabelled data points arrive in a stream and are discarded immediately or sent to an annotator. However, online AL tends to focus on maximising the accuracy of the final model (i.e. student in our terminology). In contrast, what matters in neural caching is the accuracy of the joint system (student, teacher, along with the policy) over its lifetime since this *online accuracy* reflects the average level of service offered to a user.

Despite the aforementioned differences with AL, evaluating the existing AL algorithms – specific-

ally the example selection criteria – remains valuable given the maturity of the AL field and the ease of implementation of some of the AL methods. This study aims to achieve this, as well as to investigate the potential shortcomings of these methods. For instance, will the AL methods end up selecting examples that are too challenging even for the LLM? Would learning from these noisy examples be detrimental to the student? Answering these questions can inform future research on this practically significant scenario.

In this work, our focus is specifically on classification tasks, as opposed to free text generation. Many practical problems, such as routing user requests to relevant departments or answering questions about factual knowledge, can be framed as classification tasks. By confining our focus to classification, we can apply methods developed in AL without modification. This also allows us to circumvent additional challenges tied to the automatic evaluation of text generation (Celikyilmaz et al., 2020).

Our findings reveal the benefits of using AL-based policies such as Margin Sampling (Scheffer et al., 2001) and Query by Committee (Seung et al., 1992). Across datasets and budgets, these methods consistently outperform baselines, such as routing examples randomly or training the student at the very start. Our analysis also reveals that the student appears robust to the noise introduced by an LLM. We also analyse a simplified practical scenario where the student is not retrained and observe even greater improvements in online accuracy from using AL-based policies. We release our code to encourage further work on this problem.[1]

The key contributions of this work are:

- We formulate the *neural caching* problem as a powerful extension of using static caches. In neural caching, LLM calls are optimised, while the student model is periodically retrained on the labels. We believe online Knowledge Distillation could play a key role in saving calls to expensive models.
- We release a benchmark with LLM annotations for classification tasks to facilitate future research in this setup.
- We evaluate and analyse different instance selection criteria for the neural caching setup.
- Our findings reveal that AL-based selection criteria consistently improve performance over baseline methods across various budgets and datasets.

This work is described in Ramírez et al. (2023) which is currently under review.

3.4 EEE-QA: Exploring Effective and Efficient Question-Answer Representations

Question Answering (QA) in NLP, is an area of research that aims to provide responses to questions in natural language. Like most QA systems, multiple-choice question answering is approached using pre-trained encoders. An encoder acts as a regression model to produce a score for each question-candidate answer pair indicating how suitable the answer is to the question. After all answer candidates are scored, the answer associated with the highest score is selected as the output. This requires QA inference to be run as many times as there are answer choices—we argue that this is not optimized for memory efficiency as the same question has to be encoded multiple times.

[1] https://anonymous.4open.science/r/neural-caching-780F/README.md

We investigated a method to feed a question and all answer candidates into the encoder to make a judgment in one go. First, we found that representation pooling significantly outperforms the convention which uses begin-of-sentence tokens as answer representations. This is straightforward and effective, but not widely used in QA literature. Then, we adopt a gating mechanism to enable interactions between answer representations to allow a model to discern the differences between answers. To perform question answering with N candidate choices, our work reduces memory complexity from O(N) to O(1), allowing for larger throughput and hence faster inference for a given set of questions. Practically, our work enhances 38–100% throughput with 26–65% speedups on low-end GPUs by allowing for considerably larger batch sizes.

More technical details are available in our paper accepted to LREC-COLING 2024 (Hu et al., 2024).

T6.1 Plans for future work

We plan to further investigate lexical shortlisting through KNN clustering on the output layer. We will also investigate sliding self-attention for modelling longer contexts.

4 Task T6.2: Model Compression (UEDIN)

Proposal

We will explore two different model compressions: low precision quantisation and model pruning. Low precision quantisation can be used to increase decoding speed when we use 8-bit integers. For model compression we can go further and use experimental log 4-bit quantisation which reduces the model size by a factor of 8. In addition we will explore model pruning which will remove non-essential parameters from the model. Depending how much redundancy is in the model, large blocks of it can be sparsified which will reduce model size.

The key highlights from the proposal are listed below.

- Quantisation
- Model pruning

Summary of completed work

During our work on multimodal models, we realised that representing speech in compact discrete units would not only make speech and multimodal speech and text models smaller, it would bridge the modality gap and lead to higher quality mutlimodal models. We therefore focussed our attention on this area of research.

4.1 Compact Speech Translation Models via Discrete Speech Unit Pretraining

In Speech-to-text Translation (ST), the use of Self-Supervised Leaning (SSL) models, such as wav2vec 2.0 and HuBERT (Baevski et al., 2020; Hsu et al., 2021), as model initialization is now common to obtain state-of-the-art result (Agarwal et al., 2023). Nevertheless, using these bulky

models as initialisation imposes a large memory footprint to the downstream ST, which limits flexibility in modelling choices. In addition, it hinders on-device deployment that is crucial for privacy and useful in the absence of internet connection.

Knowledge distillation (KD) (Kim and Rush, 2016; Inaguma et al., 2021) is the standard approach to creating smaller models, but obtaining the high-quality pseudo (ST) labels for training ST systems is very costly. Instead, we focus on Discrete Speech Units (DSU) since they can be obtained cheaper, e.g., without requiring a translation model, and are an intermediate representation between speech and text. DSUs are K-means clusters of speech representations from selected layers within a SSL model. It represents sequences of discrete tokens, which are easier to model within a text processing architecture (Polyak et al., 2021; Chou et al., 2023). DSU sequences are far smaller than dense Filterbank sequences, and they can be shortened with removing duplicates and Byte Pair Encoding (Sennrich et al., 2016). Therefore, in ST, a straightforward method to distill the SSL models is to replace the dense Filterbank by the DSUs, aka the DSU-to-Translation (DSU-to-Trl) model (Chang et al., 2023; Zhang et al., 2023). Although using DSUs for training allows for transfer learning, using them at inference requires large and slow models.

To address the above, we use DSU for pretraining rather than as model input to make ST models more compact. Our method leverages large SSL models by pretraining smaller models on the corresponding DSU. More specifically, it pretrains encoder-decoder models on 1) Filterbank-to-DSU (Fbk-to-DSU) and 2) DSU-to-Trl data, and takes the encoder from 1) and the decoder from 2) to initialise a new model. After that, the new model is finetuned on limited ST data. Under this formulation, (1) the model is much more compact than the SSL model (2) the compact model does not use DSU as inputs which (a) avoids the lengthy SSL and clustering inference pipeline and (b) minimizes its sensitivity to (DSU) tokenization. (3) Our method does not require transcripts, unlike ASR pretraining, making it applicable to low-resource settings. Similar to ST methods that use pretrained components, our method could be limited by the *pretraining modality gap* (Liu et al., 2020; Le et al., 2023). Motivated by prior works, we investigate mitigating it with Connectionist Temporal Classification (CTC) (Graves et al., 2006) regularisation.

We evaluate our method on CoVoST-2 (Wang et al., 2021) dataset, 21 (X-EN) language directions with end-to-end multilingual ST. By simply using a monolingual HuBERT model to extract the DSU, our method has a gain of more than 3 BLEU, average of 21 language directions, with respect to training from scratch, demonstrating DSU's cross-lingual ability. Our method is also more than 0.5 BLEU better than a ST model that directly finetuned HuBERT despite having half model size. Our main contributions are:

- We propose to distill knowledge from large SSL models to make ST models more compact via DSU pretraining. Our method avoids the lengthy inference pipeline in the DSU-to-Trl method.
- We examine the effect of (DSU) tokenization on the ST models. Our method is robust across different (DSU) tokenizations, which is not the case for the DSU-to-Trl method.
- We examine applying CTC to the DSU in pretraining and on the translation in finetuning to mitigate the modality gap, and we show that both improve performance.

The work is described in Lam et al. (2024) which is currently under review.

T6.2 Plans for future work

We plan to deploy quantisation and pruning in order to further reduce model size and make larger models run efficiently on less GPU resources.

5 Task T6.3: Providing tools for evaluation, usage, sharing and adaptation of the models (UNB, UEDIN)

Proposal highlights

Experimenting with large model checkpoints requires efficient ways of handling such large files and having a centralized repository that can handle such files. In UTTER we are leveraging open sharing and hosting platforms, namely the Hugging Face Hub[2], to store and share large datasets and models. For speech-to-text and text-to-text settings, we will develop and support: metrics for standardized evaluation, inference API for simple and efficient demo purposes, open-source scripts for generic model evaluation.

Summary of completed work

The UTTER project has been using the infrastructure provided by HuggingFace for sharing models. The so called HuggingFace hub is the "de facto" standard for sharing small and large datasets as well as large language models nowadays. UTTER has an "organization" space[3] that can centralize the storage and sharing of models and datasets developed throughout the project. This space allows not only private sharing of resources among project members but also publicly sharing such resources when they are ready for distribution. One nice by-product of having models in HuggingFace Hub and using its model format is that it is quite straightforward to perform adaptation of such models using HuggingFace code[4]. For sharing code we have a Github repository[5] for the project.

The main focus of work under this rubric has been on automating the generation and evaluation of open source large language models for different tasks, as described in section 5.1.

5.1 TowerEval

As part of the development of Tower LLM (Alves et al., 2024) we built an evaluation toolkit that can be used for several different text-based tasks, ranging from translation to grammatical error correction. The TowerEval toolkit has been implemented in Python, and it is publicly available as open source[6]. It can be used to generate outputs from any model available on the HuggingFace Hub or closed LLMs such as GPT-3.5 and GPT-4.

TowerEval provides a flexible solution to prepare the instruction data. The user can provide a template and the toolkit prepares the instructions using the raw data and the instruction template.

[2] https://huggingface.co/docs/hub/en/models-the-hub
[3] https://huggingface.co/utter-project
[4] https://huggingface.co/docs/autotrain/en/llm_finetuning
[5] https://github.com/utter-project
[6] https://github.com/deep-spin/tower-eval

The toolkit has been designed to perform a series of generation and evaluation tasks. The main requirement in order to achieve that is setting up a configuration file that points to the input data, the models to be run together with their desired settings, and the evaluation metrics to be used. The toolkit can be used for generation only or generation followed by evaluation for multiple languages.

Currently TowerEval has been used mostly for generating output and evaluating TowerLLM, and different versions of GPT for a series of generative tasks: grammar error correction, machine translation, automatic post-editing, automatic translation quality evaluation, among others. It has also been used for evaluating named entity recognition performance of the same models. The toolkit has been designed to be easily extendable to different tasks as long as they can be represented in an input prompt and the generative models are able to produce coherent output that can be parsed and compared to a gold standard annotation or reference.

One example of the toolkit's flexibility is the support to generic language understanding benchmarks like MMLU, Hellaswag, Winograde, among others. These datasets are currently the main ones used by the community to evaluate pretrained models on natural language understanding tasks. The integration of these benchmarks is achieved by a wrapper script that calls the "*lm-evaluation-harness*" toolkit[7]. The wrapper performs the generation of outputs and evaluation first, and then stores the results in the same format as all the tasks natively supported in TowerEval. This unifies the data and output structure of the evaluation tasks, and makes the evaluation process seamless to the user.

T6.3 Plans for future work

As future work we intend to extend TowerEval to cover other generative tasks such as multilingual summarization and other relevant classification or regression tasks required for different work packages in UTTER.

[7] https://github.com/EleutherAI/lm-evaluation-harness

6 Conclusion

This deliverable reports on the work done in the first half of the project on efficient and compact models and tools for evaluating large and mutimodal models. In the next half of the project, we are going to continue to investigate optimal strategies for reducing output vocabularies, combining small and large models, and expand the language and evaluation frameworks for the tools we develop to support our research.

References

Milind Agarwal, Sweta Agrawal, Antonios Anastasopoulos, Luisa Bentivogli, Ondřej Bojar, Claudia Borg, Marine Carpuat, Roldano Cattoni, Mauro Cettolo, Mingda Chen, William Chen, Khalid Choukri, Alexandra Chronopoulou, Anna Currey, Thierry Declerck, Qianqian Dong, Kevin Duh, Yannick Estève, Marcello Federico, Souhir Gahbiche, Barry Haddow, Benjamin Hsu, Phu Mon Htut, Hirofumi Inaguma, Dávid Javorský, John Judge, Yasumasa Kano, Tom Ko, Rishu Kumar, Pengwei Li, Xutai Ma, Prashant Mathur, Evgeny Matusov, Paul McNamee, John P. McCrae, Kenton Murray, Maria Nadejde, Satoshi Nakamura, Matteo Negri, Ha Nguyen, Jan Niehues, Xing Niu, Atul Kr. Ojha, John E. Ortega, Proyag Pal, Juan Pino, Lonneke van der Plas, Peter Polák, Elijah Rippeth, Elizabeth Salesky, Jiatong Shi, Matthias Sperber, Sebastian Stüker, Katsuhito Sudoh, Yun Tang, Brian Thompson, Kevin Tran, Marco Turchi, Alex Waibel, Mingxuan Wang, Shinji Watanabe, and Rodolfo Zevallos. FINDINGS OF THE IWSLT 2023 EVALUATION CAMPAIGN. In Elizabeth Salesky, Marcello Federico, and Marine Carpuat, editors, *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 1–61, Toronto, Canada (in-person and online), July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.iwslt-1.1. URL https://aclanthology.org/2023.iwslt-1.1.

Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and André F. T. Martins. Tower: An open multilingual large language model for translation-related tasks, 2024.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

Nikolay Bogoychev, Pinzhen Chen, Barry Haddow, and Alexandra Birch. Large language model inference with lexical shortlisting. In *Deployable AI (DAI) Workshop at AAAI*. 2024.

Davide Cacciarelli and Murat Kulahci. A survey on online active learning. *CoRR*, abs/2302.08893, 2023. doi: 10.48550/arXiv.2302.08893. URL https://doi.org/10.48550/arXiv.2302.08893.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. Evaluation of text generation: A survey. *CoRR*, abs/2006.14799, 2020. URL https://arxiv.org/abs/2006.14799.

Xuankai Chang, Brian Yan, Kwanghee Choi, Jeeweon Jung, Yichen Lu, Soumi Maiti, Roshan Sharma, Jiatong Shi, Jinchuan Tian, Shinji Watanabe, et al. Exploring speech recognition, translation, and understanding with discrete speech units: A comparative study. *arXiv preprint arXiv:2309.15800*, 2023.

Ju-Chieh Chou, Chung-Ming Chien, Wei-Ning Hsu, Karen Livescu, Arun Babu, Alexis Conneau, Alexei Baevski, and Michael Auli. Toward joint language modeling for speech units and text. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6582–6593, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.438. URL https://aclanthology.org/2023.findings-emnlp.438.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In William W. Cohen and Andrew W. Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 369–376. ACM, 2006. doi: 10.1145/1143844.1143891. URL https://doi.org/10.1145/1143844.1143891.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Zhanghao Hu, Yijun Yang, Junjie Xu, Yifu Qiu, and Pinzhen Chen. EEE-QA: Exploring effective and efficient question-answer representations, 2024. URL https://arxiv.org/abs/2403.02176. Accepted to the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation.

Hirofumi Inaguma, Tatsuya Kawahara, and Shinji Watanabe. Source and target bidirectional knowledge distillation for end-to-end speech translation. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1872–1881, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.150. URL https://aclanthology.org/2021.naacl-main.150.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In Jian Su, Kevin Duh, and Xavier Carreras, editors, *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.

Tsz Kin Lam, Alexandra Birch, and Barry Haddow. Compact speech translation models via discrete speech units pretraining, 2024.

Phuong-Hang Le, Hongyu Gong, Changhan Wang, Juan Pino, Benjamin Lecouteux, and Didier Schwab. Pre-training for speech translation: CTC meets optimal transport. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 18667–18685. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/le23a.html.

Yuchen Liu, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. Bridging the modality gap for speech-to-text translation. *arXiv preprint arXiv:2010.14920*, 2020.

Adam Polyak, Yossi Adi, Jade Copet, Eugene Kharitonov, Kushal Lakhotia, Wei-Ning Hsu, Abdelrahman Mohamed, and Emmanuel Dupoux. Speech Resynthesis from Discrete Disentangled Self-Supervised Representations. In *Proc. Interspeech 2021*, pages 3615–3619, 2021. doi: 10.21437/Interspeech.2021-475.

Guillem Ramírez, Matthias Lindemann, Alexandra Birch, and Ivan Titov. Cache & distil: Optimising api calls to large language models, 2023.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active Hidden Markov Models for Information Extraction. In Frank Hoffmann, David J. Hand, Niall Adams, Douglas Fisher, and Gabriela Guimaraes, editors, *Advances in Intelligent Data Analysis*, Lecture Notes in Computer Science, pages 309–318, Berlin, Heidelberg, 2001. Springer. ISBN 978-3-540-44816-7. doi: 10.1007/3-540-44816-0_31.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. *Commun. ACM*, 63(12): 54–63, 2020.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162.

Burr Settles. Active Learning Literature Survey. Technical Report, University of Wisconsin-Madison Department of Computer Sciences, 2009. URL https://minds.wisconsin.edu/handle/1793/60660. Accepted: 2012-03-15T17:23:56Z.

H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294, Pittsburgh Pennsylvania USA, July 1992. ACM. ISBN 978-0-89791-497-0. doi: 10.1145/130385.130417. URL https://dl.acm.org/doi/10.1145/130385.130417.

Marcos Treviso, Ji-Ung Lee, Tianchu Ji, Betty van Aken, Qingqing Cao, Manuel R Ciosici, Michael Hassid, Kenneth Heafield, Sara Hooker, Colin Raffel, et al. Efficient methods for natural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 11:826–860, 2023.

Changhan Wang, Anne Wu, Jiatao Gu, and Juan Pino. CoVoST 2 and Massively Multilingual Speech Translation. In *Proc. Interspeech 2021*, pages 2247–2251, 2021. doi: 10.21437/Interspeech.2021-2027.

Xueying Zhan, Qingzhong Wang, Kuan-Hao Huang, Haoyi Xiong, Dejing Dou, and Antoni B. Chan. A comparative survey of deep active learning. *CoRR*, abs/2203.13450, 2022. doi: 10.48550/arXiv.2203.13450. URL https://doi.org/10.48550/arXiv.2203.13450.

Dong Zhang, Rong Ye, Tom Ko, Mingxuan Wang, and Yaqian Zhou. DUB: Discrete unit back-translation for speech translation. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7147–7164, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.447. URL https://aclanthology.org/2023.findings-acl.447.

ENDPAGE

UTTER

HORIZON-CL4-2021-HUMAN-01 101070631

D6.1 First report on efficient inference