Multilingual Simultaneous Neural Machine Translation

Philip Arthur[†*] **Dongwon K. Ryu**[‡*] **Gholamreza Haffari**[‡]

[†]Oracle Digital Assistant, Oracle
[‡]Department of Data Science and AI, Monash University
philip.arthur@oracle.com
{dongwon.ryu, gholamreza.haffari}@monash.edu

Abstract

Simultaneous machine translation (SIMT) involves translating source utterances to the target language in real-time before the speaker utterance completes. This paper proposes the *multilingual* approach to SIMT, where a single model simultaneously translates between multiple language-pairs. This not only results in more efficiency in terms of the number of models and parameters (hence simpler deployment), but may also lead to higher performing models by capturing commonalities among the languages. We further explore simple and effective multilingual architectures based on two strong recently proposed SIMT models. Our results on translating from two Germanic languages (German, Dutch) and three Romance languages (French, Italian, Romanian) into English show (i) the single multilingual model is on-par or better than individual models, and (ii) multilingual SIMT models trained based on language families are on-par or better than the universal model trained for all languages.[1]

1 Introduction

Simultaneous translation is the task of incrementally generating the translation while the source utterance is gradually spoken. It is crucial in multinational meetings, e.g., in business and politics, where the online simultaneous translation is required for one or multiple language-pairs. Simultaneous machine translation (SIMT) is an attempt to address the challenges of this translation scenario, i.e., trading off the translation quality and its latency (Cho and Esipova, 2016; Arivazhagan et al., 2019, 2020; Firat et al., 2016b).

In this paper, we investigate the *multilingual* SIMT setting, where a single model simultaneously translates between multiple language-pairs. This not only results in more efficiency in terms of the number of models and parameters (hence simpler deployment), but may also lead to higher performing models by capturing commonalities among the languages. The multilingual setting has been successful for the standard offline neural machine translation (NMT) and studied extensively (Johnson et al., 2017; Tan et al., 2019; Aharoni et al., 2019).

We explore simple and effective multilingual architectures based on two strong recently proposed SIMT models, i.e. the WAIT-K (Dalvi et al., 2018) and COUPLED POLICY (Arthur et al., 2020). The former waits to *read* a fixed number of k input tokens; afterward, it *writes* (generates) an output token for each newly received input token. The latter learns a policy, via an *agent*, for an adaptive waiting between reading and writing to reduce the translation delay while maintaining the quality.

COUPLED POLICY uses the adaptive waiting, generated from offline word alignments. It continues to read the source tokens until the corresponding word alignment to the target token appears.

Under these underlying SIMT models, we explore multi-task learning (MTL) framework, *full* and *partial* parameter sharing protocols across the languages with language indicators.

Our experiments show the effectiveness of the simple strategy of sharing all the SIMT components across the languages, with language tags specifying the translation task. The results on translating from two Germanic languages (German, Dutch) and three Romance languages (French, Italian, Romanian) into English show the single multilingual model is on-par or better than individual models. Furthermore, the results show that multilingual SIMT models trained based on language families are on-par or better than the universal model trained for all languages.

[1] Star (*) marks a shared first authorship between Philip and Dongwon, where both contributed equally. This work was done when Philip was a research fellow at Monash University.

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4758–4766
August 1–6, 2021. ©2021 Association for Computational Linguistics

Algorithm 1 Training Multilingual NPI-SiMT

```
Require: $\mathcal{D}$: Collections of parallel corpora with
    oracle actions.
while a stopping condition is not met do
    for $\mathcal{D}_i \in \mathcal{D}$ do
        $F, E$ is a language pair of $\mathcal{D}_i$.
        for $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}) \in \mathcal{D}_i$ do
            $\theta_{\mathcal{A}}^{F,E} \leftarrow \text{MLE}(\boldsymbol{a}, \theta_{\mathcal{A}}^{F,E})$
            $\theta_{\mathcal{E}}^{F}, \theta_{\mathcal{D}}^{E} \leftarrow \text{MLE}(\boldsymbol{x}, \boldsymbol{y}, \theta_{\mathcal{E}}^{F}, \theta_{\mathcal{D}}^{E})$
        end for
    end for
end while
```

2 Multilingual Simultaneous Translation

The original neural programmer-interpreter (NPI)-SiMT framework (Arthur et al., 2020) employs a trainable programmer θ_{prog} and interpreter θ_{intp}. The programmer/agent issues *read*/*write* commands to control the interpreter, i.e. the NMT model. The interpreter is constructed using an encoder $\theta_{\mathcal{E}}$ and a decoder $\theta_{\mathcal{D}}$. Each component is trained on triplet $\langle x, y, a\rangle$ where x is the source sentence, y is the target sentence, and a is the program oracle using behavioral cloning (Torabi et al., 2019). For notation clarity we rename the programmer into $\theta_{\mathcal{A}}$, resulting triplet of trainable modules $\theta_{\mathcal{A}}, \theta_{\mathcal{E}}, \theta_{\mathcal{D}}$.

Language-Specific Parameters We further extend this framework by distilling a parameter, specific to language θ_x^l, where x is a specific module and l is a specific language. This language-specific parameter is similar to Firat et al. (2016a); Dong et al. (2015); Ahmadnia and Dorr (2020) where parameters are separated based on the source and target languages. In the case of SiMT, the program a is affected by both languages. This framework enables us to use multiple parallel corpora $\mathcal{D}_i$ and train a language specific module using maximum likelihood estimation by updating particular θ_x^l based on $\mathcal{D}_i$. The training algorithm for our NPI-SiMT is shown in Algorithm 1.

Multilingual Parameter Sharing Multilingual parameter sharing is achieved by using only a single module for language-specific parameter θ_x^*. Depending on the module, we can disregard source (F) or target (E) completely. This allows us to share the parameter across different parallel corpora. However, the embedding matrix in different $\mathcal{D}_i$ can be different because of various tokenization and vocabulary construction methods. To remedy this, we can either train joint vocabulary spaces for source and target sides, or simply joining different spaces using union operation. Herein, we use the latter method.

Language Indicator Embedding When the interpreter is shared, it is difficult to communicate which pairs of languages are being processed. To deliver this, we pass the source and target language embedding information to the encoder and decoder, respectively. This information is then combined using addition operation with the word embedding. In the programmer, we use a concatenation of both source and target languages.

Batch of Multilingual Instances Algorithm 1 outlines the overall training procedure of multilingual SiMT. Here, it is crucial to construct a batch as a mixture of many language pairs to achieve good multilingual training. We also need to include the information of the source language to create language indicator embedding. If a module is language-agnostic, it will be responsible for consuming all the input; otherwise, language-specific modules will be used to process the specific item in the batch according to its language. Results from different languages will be aggregated using concatenation at the end.

3 Experiments

Our experiment aims to investigate the effects of multilingualism in SiMT architecture. To achieve this we first choose the language pairs from (1) the same family group and (2) mixing them all. This is enabled by investigating various parameter sharing strategies for the components of the SiMT architectures.

Datasets. We use IWSLT 2017 (Cettolo et al., 2017) datasets for all parallel corpora in which all translating to English. The choice was made due to its characteristics of spoken multilingual corpora from TED. We choose the Germanic language group, German (DE) and Dutch (NL), and the Romance language group, Italian (IT), French (FR), and Romanian (RO). The languages within the same group generally have high syntactic similarity and the same word order. Unless otherwise specified, we use the same settings and preprocessing as described in Arthur et al. (2020).[2]

[2] In our preliminary experiment, the pre-processing under concatenation of multilingual corpora with larger vocabulary

SiMT Systems We compared two SiMT baselines, Coupled Policy (Arthur et al., 2020) and the Wait-k model (Ma et al., 2018). For a fair comparison, we choose a value of k, which achieves comparable translation quality to the Coupled Policy system. Following some initial experiments, we choose $k = 2$.

Parameter Sharing Since our model deals with the many-to-one translation task with an agent, we decided to separate i) encoder, ii) agent, and iii) encoder + agent. This idea came from the performance improvements that a number of studies demonstrated by separating the decoder in offline one-to-many MT (Dong et al., 2015; Sachan and Neubig, 2018). In SiMT, two modules, encoder and agent, are tied to the source, and therefore, reasonable to have them as language-specific parameters.

Evaluation. Following Arthur et al. (2020), we evaluate the systems based on their translation quality and delay. Translation quality can be measured by case sensitive BLEU (Papineni et al., 2002). [3] We adopt two delay measurements by previous studies: (1) average proportion (AP) (Cho and Esipova, 2016) is a fraction of reading source words per emitted target words, and (2) average lagging (AL) (Ma et al., 2019) is an average number of lagged source words until all inputs are read.

3.1 Results

In this section, we will describe the results of parameter sharing in SiMT. Following that, we present the comparison of multilingualism under different language groups.

Parameter Sharing Strategies. Table 1 presents the results of various parameter sharing strategies for FR/IT/RO in the Romance language family. When sharing all parameters across these three languages, Wait-2 has a slight increase in delay, but the translation quality is comparable to or better than bilingual. In contrast, the best parameter sharing setting for Couple Policy is to have language-specific encoders and share the rest of the parameters. This appears to have a clear advantage in both quality and delay; the BLEU score increases up to 0.8 units, with a reduction in AL, approximately 10% in FR and IT. In both architectures, the model size reduces drastically when trained on multilingual setting, and remains approximately the same across different sharing strategies. These results are consistent for DE/NL in the Germanic language family. Full results are included in the supplementary material.

			FR→EN		IT→EN		RO→EN		Model
$\theta_{\mathcal{E}}$	$\theta_{\mathcal{D}}$	$\theta_{\mathcal{A}}$	AL	BLEU	AL	BLEU	AL	BLEU	Size
				Wait-2					
×	×	–	**2.22**	23.40	**2.73**	24.77	**2.55**	24.34	158.7M
×	✓	–	2.32	23.54	2.77	24.75	2.68	**24.43**	88.2M
✓	✓	–	2.35	**23.76**	2.77	**25.24**	2.72	24.10	84.1M
				Coupled Policy					
×	×	×	1.48	26.90	1.00	22.62	1.02	22.82	166.9M
×	✓	×	1.53	24.82	0.90	21.13	1.09	20.60	96.4M
×	✓	✓	**1.37**	**27.45**	0.92	**23.40**	**1.01**	**23.35**	90.9M
✓	✓	×	1.44	27.36	0.99	23.13	1.16	23.00	92.4M
✓	✓	✓	1.38	27.34	**0.89**	23.32	1.02	22.63	86.9M

Table 1: Parameter sharing under Romance language family: French, Italian, Romanian. ✓ and × indicate *shared* and *not sharing* components in the MTL architecture.

Multilingual Modelling Strategies. Table 2 shows the overall performance comparison of the multilingual setting. Multilingualism in SiMT evidently surpasses the bilingual baseline in translation delay, quality, and/or model size. Generally, SiMT trained on the same language family outperforms not only the bilingual baselines, but also the universal multilingual model. In the Germanic language, training under the same language group boosts up the BLEU up to 1 unit. Although baseline in Wait-2 has a shorter delay, Coupled Policy surpasses both quality and delay. We observe that when the model runs universally, the BLEU score reaches back to or lower than that of the bilingual model.

On the other hand, the Romance language family has slightly different behavior across different SiMT models. Coupled Policy behaves similarly, where training SiMT under the same group positively influences the performance, but in Wait-2, the universal model excels the best. This is particularly interesting because the Romance language family has the same word order as English, which Wait-2 would be a perfect fit for such translation between two languages with the same word order. However, it is not the case, so mixing all the languages regardless of word order under Wait-2 improves translation quality more while preserving the delay.

size does not impact performance.

[3] Calculated using sacrebleu (Post, 2018). BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.4 is our `sacreblue`'s signature

	DE→EN			NL→EN			FR→EN			IT→EN			RO→EN			
	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU	Model Size
WAIT-2																
Bilingual	0.62	**2.54**	22.99	0.63	**2.39**	28.33	0.60	2.22	23.40	0.63	2.73	24.77	0.63	2.55	24.34	264.5M
Germanic	**0.62**	2.56	**23.86**	**0.63**	2.50	**28.44**	–	–	–	–	–	–	–	–	–	68.5M
Romance	–	–	–	–	–	–	0.61	2.35	23.76	0.64	2.77	**25.24**	0.63	2.72	24.10	84.1M
Universal	0.62	2.55	23.04	0.63	2.44	27.43	**0.60**	**2.21**	**24.40**	**0.63**	**2.58**	25.09	**0.63**	**2.54**	**24.63**	115.4M
COUPLED POLICY																
Bilingual	0.56	1.55	21.33	0.56	1.35	26.63	0.55	1.48	26.90	0.53	1.00	22.62	0.54	1.02	22.82	278.2M
Germanic	**0.56**	**1.47**	**22.33**	**0.55**	1.26	**27.25**	–	–	–	–	–	–	–	–	–	73.3M
Romance	–	–	–	–	–	–	**0.55**	**1.37**	**27.45**	**0.53**	**0.92**	**23.40**	**0.54**	**1.01**	**23.35**	90.9M
Universal	0.56	1.57	21.33	0.55	**1.24**	26.03	0.55	1.37	27.27	0.54	0.97	23.17	0.54	1.04	22.44	118.2M

Table 2: Multilingual results for WAIT-2 and COUPLED POLICY under Bilingual, Universal, and the same language family (Germanic and Romance). Fully shared architecture is selected for Universal model while the same language group model has language-specific parameters for encoder. The last column indicates model parameter size, where bilingual row adds up the model size of all the language pairs, i.e. $55.646\text{M} \times 5 \approx 278.2\text{M}$.

Under the same SIMT model, COUPLED POLICY has better performance when trained in the same language group. Also, the model size decreases 40% compared to the bilingual baseline, where the same language family has a total of 164.2M parameters and the bilingual has a total of 278.2M parameters. WAIT-2 seems to have slightly arguable results, where DE and NL have the highest BLEU when trained language-family-wise, but the Romance language family benefits the most from universally trained in all languages. Also, one should note that a lower delay in WAIT-K under the same k value does not mean outputting the target sentence faster: (1) Because of the nature of WAIT-K, the model follows the fixed READ and WRITE actions, and (2) the formulation of AL accounts for not only the lagging of translation but also the number of tokens produced as output and taken as input. Therefore, a lower AL indicates the changes in the probability of producing the end of the sentence. This will generate a shorter target sentence and/or stop the translation without fully observing the input, which impacts the delay. Nevertheless, under WAIT-2, the translation quality improves, and the model size decreases with multilingualism.

3.2 Discussion

The setting for parameter sharing in this experiment is inspired from the observation that the multilingual NMT can benefit from separating encoder and decoder parameters (Dong et al., 2015; Sachan and Neubig, 2018; Ahmadnia and Dorr, 2020). The motivation from Dong et al. (2015); Sachan and Neubig (2018) is that separating decoder parameters in one-to-many setting is beneficial because of the difficulty of one-to-many translation task. Our problem is SIMT where not only mapping from the source language to the target language is important, but also learning when to map is equally important. Hence, our assumption was that due to the difficulty of many-to-one SIMT task, assigning the encoder and the agent to language-specific would help the performance. Under WAIT-K, encoding the representation of the source language separately does not seem to benefit. However, Table 2 shows multilingual setting surpasses bilingual. This would be similar to the traditional NMT, that the model generalize the translation tasks across different languages and leverages the correlation across the source languages.

COUPLED POLICY is more complex architecture than WAIT-K as it also needs to learn the optimal policy from an oracle trajectory. However, this takes more advantages when trained on the same language family. Since its oracle is generated from offline word alignments between the source language and the target language, its mechanism of read/write is dependent on the word order and language properties. Our results in Table 2 also supports this as the model trained on the same language family surpasses both bilingual baselines and universal model. The interesting observation here is that, unlike WAIT-K, a separated encoder takes advantages more than fully shared architecture while separating both encoder and agent significantly degrades the performance in BLEU. This suggests that the language-specific encoder can form the representation of the source languages better than the shared one, but if the agent is separated together, the model struggles mapping from the source language to the target language. This reflects why separated encoder and agent in COUPLED POLICY has a BLEU decay while AL is not affected as significant. Therefore, because COUPLED POLICY takes advantages of the same word order from its oracle trajectory, its shared agent can capture the general representation of the same word order better while the language-specific encoder can help the agent by only focusing on encoding the representation of each source language.

4 Related Work

Simultaneous Machine Translation SIMT has been explored as a sequential decision-making translation problem. NPI architecture is employed to 1) choose whether to take more input token or produce output token using agent programmer and 2) translate partially observed input tokens to output using neural machine translation (NMT) interpreter (Satija and Pineau, 2016; Gu et al., 2016). The initial approaches were mainly training the agent using reinforcement learning with assigned rewards to balance the trade-off between translation quality and delay (Gu et al., 2016; Satija and Pineau, 2016; Alinejad et al., 2018). However, it has stability and robustness issues due to the sparse reward signals, so imitation learning using oracle actions has been independently attempted (Zheng et al., 2019; Arthur et al., 2020; Dalvi et al., 2018).

Multilingual Machine Translation In NMT, multilingual training is a popular MTL approach as it is very simple, but effective (Johnson et al., 2017; Sachan and Neubig, 2018; Dong et al., 2015; Dabre et al., 2020). Instead of choosing entirely different NLP tasks and increase complexity of implementation (Niehues and Cho, 2017; Zaremoodi and Haffari, 2018), multilingual setting only involves concatenating multiple bilingual language pairs for training (Johnson et al., 2017). The language pairs are the task space in MTL, which determines the performance of the model, and so, the selection of language pairs influences the overall performance of translation (Tan et al., 2019).

Parameter sharing in multilingual setting has also been extensively studied. Dong et al. (2015) initially had language-specific decoder under one-to-many translation. This was further extended to sharing decoder parameters partially (Sachan and Neubig, 2018). Ahmadnia and Dorr (2020) investigated hierarchically sharing parameters under the similarity between languages. This simple parameter sharing has shown to restrict sharing dissimilarity, improving translation quality of all the languages (Johnson et al., 2017; Sachan and Neubig, 2018; Cettolo et al., 2017).

5 Conclusions

In this paper, we have investigated multilingual SIMT using IWSLT 2017 datasets. We have explored simple and effective multilingual architectures based on two strong recently proposed SIMT models, namely WAIT-K and COUPLED POLICY. Experiments show that the best parameter sharing strategy for the WAIT-K model, when dealing with DE/NL (as Germanic languages) and RO/IT/FR (as Romance languages), is to share all SIMT components across the languages regardless of the language set. However, the best sharing strategy seems to depend on the language family when it comes to COUPLED POLICY. Under the best parameter sharing strategy, our results have shown that (i) the single multilingual model is on-par or better than individual models, and (ii) multilingual SIMT models trained based on language families are on-par or better than the universal model trained for all languages. Furthermore, (iii) COUPLED POLICY takes the advantages of the same word order, so it achieves the best performance with the language-specific encoder and training under the same language family

For the future work, we plan to extend this to a larger dataset. Aharoni et al. (2019) demonstrated the scales of parallel corpora draw different conclusions in multilingual NMT. To maintain the characteristics of spoken languages, translation datasets must be selected carefully. Secondly, we will investigate different language families, including Slavic languages and Austronesian languages. The consistent results in different families would make the claim in this paper more valid.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Benyamin Ahmadnia and Bonnie Dorr. 2020. A new approach to parameter-sharing in multilingual neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (AMTA 2020)*, pages 1–6, Virtual. Association for Machine Translation in the Americas.

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Te I, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020. Re-translation strategies for long form, simultaneous, spoken language translation.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1313–1323, Florence, Italy. Association for Computational Linguistics.

Philip Arthur, Trevor Cohn, and Gholamreza Haffari. 2020. Learning coupled policies for simultaneous machine translation.

M. Cettolo, M. Federico, L. Bentivogli, Niehues Jan, Stüker Sebastian, Sudoh Katsuitho, Yoshino Koichiro, and Federmann Christian. 2017. Overview of the iwslt 2017 evaluation campaign.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *CoRR*, abs/1606.02012.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *ACM Comput. Surv.*, 53(5).

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 493–499, New Orleans, Louisiana. Association for Computational Linguistics.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California. Association for Computational Linguistics.

Orhan Firat, Baskaran Sankaran, Yaser Al-onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016b. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 268–277, Austin, Texas. Association for Computational Linguistics.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O. K. Li. 2016. Learning to translate in real-time with neural machine translation. *CoRR*, abs/1610.00388.

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. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Qingsong Ma, Ondřej Bojar, and Yvette Graham. 2018. Results of the WMT18 metrics shared task: Both characters and embeddings achieve good performance. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 671–688, Belgium, Brussels. Association for Computational Linguistics.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. *CoRR*, abs/1708.00993.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Devendra Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 261–271, Brussels, Belgium. Association for Computational Linguistics.

Harsh Satija and Joelle Pineau. 2016. Simultaneous machine translation using deep reinforcement learning.

Xu Tan, Jiale Chen, Di He, Yingce Xia, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with language clustering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 963–973, Hong Kong, China. Association for Computational Linguistics.

Faraz Torabi, Garrett Warnell, and Peter Stone. 2019. Recent advances in imitation learning from observation.

Poorya Zaremoodi and Gholamreza Haffari. 2018. Neural machine translation for bilingually scarce scenarios: a deep multi-task learning approach. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1356–1365, New Orleans, Louisiana. Association for Computational Linguistics.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019. Simultaneous translation with flexible policy via restricted imitation learning. *CoRR*, abs/1906.01135.

A Appendices

SIMT Architecture We used a single layer long short term memory (LSTM) SEQ2SEQ as the interpreter for both SIMT models. The programmer is an LSTM transducer with a binary softmax to generate a read or write action for COUPLED POLICY. WAIT-K followed the fixed oracle actions under $k = 2$ without the programmer. We used scheduled sampling of 5%, 15%, and 15% for training the NPI-SIMT framework as described in Arthur et al. (2020).

Training is done using Adam (Kingma and Ba, 2015) with initial learning rate 0.001, and halved it each time when perplexity increased on the development set. Early stopping is reached at the fourth learning rate. During testing we use a standard beam search algorithm similar to Gu et al. (2016) with $b = 5$. Training is done using single V-100 GPU for 6 hours for one source language. Multilingual experiments time scale linearly to the numbers of parallel corpora being used for the experiment.

			DE→EN			NL→EN		
$\theta_{\mathcal{E}}$	$\theta_{\mathcal{D}}$	$\theta_{\mathcal{A}}$	AP	AL	BLEU	AP	AL	BLEU
WAIT-2								
×	×	–	0.62	**2.54**	22.99	0.63	**2.39**	28.33
×	✓	–	0.63	2.67	23.04	0.63	2.51	28.16
✓	✓	–	**0.62**	2.56	**23.86**	**0.63**	2.50	**28.44**
COUPLED POLICY								
×	×	×	0.56	1.55	21.33	0.56	1.35	26.63
×	✓	×	0.56	1.66	21.61	0.55	1.32	25.65
×	✓	✓	**0.56**	**1.47**	**22.33**	**0.55**	**1.26**	**27.25**
✓	✓	×	0.56	1.68	21.21	0.55	1.28	25.63
✓	✓	✓	0.56	1.49	21.69	0.56	1.31	26.22

Table 3: Parameter sharing under Germanic language family: German, Dutch. ✓ and × indicate *shared* and *not sharing* components in the MTL architecture.

			FR→EN			IT→EN			RO→EN		
$\theta_{\mathcal{E}}$	$\theta_{\mathcal{D}}$	$\theta_{\mathcal{A}}$	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU
WAIT-2											
×	×	–	**0.60**	**2.22**	23.40	**0.63**	**2.73**	24.77	0.63	**2.55**	24.34
×	✓	–	0.61	2.32	23.54	0.64	2.77	24.75	0.63	2.68	**24.43**
✓	✓	–	0.61	2.35	**23.76**	0.64	2.77	**25.24**	**0.63**	2.72	24.10
COUPLED POLICY											
×	×	×	0.55	1.48	26.90	0.53	1.00	22.62	0.54	1.02	22.82
×	✓	×	0.55	1.53	24.82	0.53	0.90	21.13	0.54	1.09	20.60
×	✓	✓	**0.55**	**1.37**	**27.45**	**0.53**	0.92	**23.40**	**0.54**	**1.01**	**23.35**
✓	✓	×	0.55	1.44	27.36	0.53	0.99	23.13	0.54	1.16	23.00
✓	✓	✓	0.55	1.38	27.34	0.53	**0.89**	23.32	0.54	1.02	22.63

Table 4: Parameter sharing under Romance language family: French, Italian, Romanian. ✓ and × indicate *shared* and *not sharing* components in the MTL architecture.

			DE→EN			FR→EN			IT→EN			NL→EN			RO→EN		
$\theta_{\mathcal{E}}$	$\theta_{\mathcal{D}}$	$\theta_{\mathcal{A}}$	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU	AP	AL	BLEU
WAIT-2																	
×	×	–	0.62	**2.54**	22.99	0.60	2.22	23.40	0.63	2.73	24.77	0.63	**2.39**	**28.33**	0.63	2.55	24.34
×	✓	–	0.62	2.59	22.85	0.60	2.22	24.04	0.63	2.60	**25.91**	0.63	2.48	27.96	0.63	2.54	24.47
✓	✓	–	**0.62**	2.55	**23.04**	**0.60**	**2.21**	**24.40**	**0.63**	**2.58**	25.09	**0.63**	2.44	27.43	**0.63**	**2.54**	**24.63**
COUPLED POLICY																	
×	×	×	0.56	1.55	21.33	0.55	1.48	26.90	0.53	1.00	22.62	0.56	1.35	**26.63**	0.54	**1.02**	**22.82**
×	✓	×	0.57	1.76	19.54	0.55	1.45	27.04	0.54	1.04	22.29	0.56	1.36	24.50	0.54	1.09	22.34
×	✓	✓	**0.55**	**1.29**	19.95	0.55	**1.33**	25.50	**0.53**	**0.92**	21.73	0.56	1.29	24.89	0.54	1.05	21.72
✓	✓	×	0.56	1.67	21.11	0.55	1.42	26.92	0.53	0.99	22.36	0.56	1.37	25.31	0.54	1.09	22.68
✓	✓	✓	0.56	1.57	**21.33**	**0.55**	1.37	**27.27**	0.54	0.97	**23.17**	**0.55**	**1.24**	26.03	**0.54**	1.04	22.44

Table 5: Parameter sharing under all language families: German, French, Italian, Dutch, Romanian. ✓ and × indicate *shared* and *not sharing* components in the MTL architecture.

Model Name	Model Size
`model.src_embedder.0.0.weight`	`torch.Size([32004, 512])`
`model.trg_embedder.1.0.weight`	`torch.Size([32004, 512])`
`model.encoder.0.weight_ih_l0`	`torch.Size([2048, 512])`
`model.encoder.0.weight_hh_l0`	`torch.Size([2048, 512])`
`model.encoder.0.bias_ih_l0`	`torch.Size([2048])`
`model.encoder.0.bias_hh_l0`	`torch.Size([2048])`
`model.decoder.1.weight_ih_l0`	`torch.Size([2048, 1024])`
`model.decoder.1.weight_hh_l0`	`torch.Size([2048, 512])`
`model.decoder.1.bias_ih_l0`	`torch.Size([2048])`
`model.decoder.1.bias_hh_l0`	`torch.Size([2048])`
`model.attention.1.src_projector.weight`	`torch.Size([512, 512])`
`model.attention.1.trg_projector.weight`	`torch.Size([512, 512])`
`model.attention.1.inner_projector.1.weight`	`torch.Size([1, 512])`
`model.context_projector.1.weight`	`torch.Size([512, 1024])`
`model.context_projector.1.bias`	`torch.Size([512])`
`model.output_projector.1.weight`	`torch.Size([32004, 512])`
`model.output_projector.1.bias`	`torch.Size([32004])`
`agent.action_embedder.0.weight`	`torch.Size([6, 512])`
`agent.input_projector.0.weight`	`torch.Size([512, 1536])`
`agent.input_projector.0.bias`	`torch.Size([512])`
`agent.rnn.0.weight_ih_l0`	`torch.Size([2048, 512])`
`agent.rnn.0.weight_hh_l0`	`torch.Size([2048, 512])`
`agent.rnn.0.bias_ih_l0`	`torch.Size([2048])`
`agent.rnn.0.bias_hh_l0`	`torch.Size([2048])`
`agent.output_projector.0.weight`	`torch.Size([6, 512])`
`agent.output_projector.0.bias`	`torch.Size([6])`

Table 6: The details of parameters in COUPLED POLICY baseline. WAIT-K has no `agent` parameters and multilingual models has additional components in `model.encoder` or `agent` and an increase in the first dimension of `model.src_embedder` by the number of languages, e.g., `torch.Size([32004, 512])` for bilingual baseline to `torch.Size([64004, 512])` for Germanic family. Total number of parameters for encoder, agent, attention, decoder, input embedding, output embedding and action embedding are 2.0M, 2.8M, 512.5K, 19.1M, 15.6M, 15.6M and 2.0K, respectively.