Neuron-Level Language Tag Injection Improves Zero-Shot Translation Performance

Jay Orten, Ammon Shurtz, Nancy Fulda, Stephen D. Richardson
Brigham Young University, USA
{jo288, acshurtz, nfulda, srichardson}@byu.edu

Abstract

Language tagging, a method whereby source and target inputs are prefixed with a unique language token, has become the de facto standard for conditioning Multilingual Neural Machine Translation (MNMT) models on specific language directions. This conditioning can manifest effective zero-shot translation abilities in MT models at scale for many languages. Expanding on previous work, we propose a novel method of language tagging for MNMT, *injection*, in which the embedded representation of a language token is concatenated to the input of every linear layer. We explore a variety of different tagging methods, with and without injection, showing that injection improves zero-shot translation performance with up to a 2+ BLEU score point gain for certain language directions in our dataset.

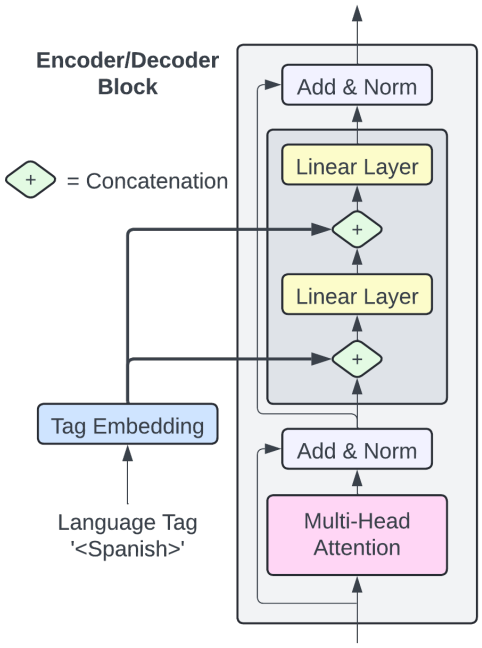

Figure 1: Language tags are injected at the neuron-level by concatenating their embedding vector to the input of the linear layers in the encoder and decoder blocks.

1 Introduction

An exciting advantage of Multilingual Neural Machine Translation (MNMT) systems is the ability for transfer learning to occur from supervised language pairs to unsupervised, zero-shot language pairs (Johnson et al., 2017; Pham et al., 2019; Gu et al., 2019). These systems enable a simplified training approach, because only a single model is necessary for any number of languages. Furthermore, because a single representation space is shared across all languages, performance is boosted for low-resource languages and training data is not required for every possible pair (Firat et al., 2016; Ha et al., 2016). This approach has been shown to scale up to over 100 languages (Aharoni et al., 2019; Fan et al., 2021).

In MNMT tasks, a common training approach includes using language tags to signify source and target language directions in the translation pair (Johnson et al., 2017). Such a tag is inserted into the model input, whereby it is operated on by the multi-headed attention mechanisms present in the encoder and decoder (Ha et al., 2016). Thus, the language direction representations within the model are learned implicitly by the optimization algorithm.

Language tagging has proven to be very effective across many tasks, and several approaches have been tested; for example, tags can be inserted on the source side, target side, or both (Wicks and Duh, 2022), and the format of the tag can vary (Blackwood et al., 2018). However, it remains unclear which tagging strategies are best suited for certain tasks, and to what extent the tag information is propagated throughout the network.

Previous research has investigated neuron-level control codes (Orten and Fulda, 2025), whereby the embedding of some conditioning information

is concatenated with the input of each feed-forward layer in the encoder and decoder blocks. In this manner, the embedded representation is directly distributed into every layer of the model. We expand upon this research by applying it to the challenging domain of zero-shot translation. Specifically, we use neuron-level injection with language tags to improve translation performance, as shown in Figure 1.

The primary contributions of this work are as follows:

- We propose a novel tagging method for MNMT models, *injection*, where embedded representations of source and target language tags are directly concatenated with the inputs into linear layers of the encoder and decoder.
- We compare our method to four existing tagging approaches and show that, for each approach, there is a method of injection that improves on the prompt-only approach, sometimes up to 2+ BLEU score points on certain language pairs.
- To test the robustness of injection, we conduct several ablation tests, showing that, despite variations in model dimensions, the injection method always performs better on average over prompt-only language tagging, specifically in regard to unseen zero-shot pairs.

2 Related Works

Language tagging has become a common method for specifying language direction in MNMT tasks (Dabre et al., 2020). Ha et al. (2016) proposed prompt tagging with their introduction of a universal encoder and decoder architecture for all training languages. They utilized unique textual tags for language-specific coding to ensure a desired target language as output. Johnson et al. (2017) achieved state-of-the-art results with zero-shot translation by including an artificial token in the beginning of input sentences. The vast improvements observed by these approaches allow a single MNMT system to scale to over 100 languages, potentially capable of translating between thousands of language pairs, without the need for each language pair to have dedicated training data.

Previous studies have investigated the impact of different tagging strategies on model performance. Wu et al. (2021) studied the impact of four different prompt-only tagging strategies on zero-shot pairs, finding that including the target tag in the encoder increased performance significantly over other methods. Their findings suggest that the target language tag is more important than the source language tag. In contrast, N ElNokrashy et al. (2022) tested including both source and target tags in the encoder and the target tag only in the decoder, finding that the inclusion of the source signal conditions the model more explicitly, reducing confusion in non-English-centric cases. Finally, Wicks and Duh (2022) investigated several methods for language token prefixing, concluding that, while the correct tagging strategy depends on the language set, source-side tag prefixes can consistently improve performance; however, they primarily focus their tests on supervised settings.

Previous research by Orten and Fulda (2025) applied control codes at the neuron level, similar to our injection method, in order to achieve improved controlled text generation. However, this research only tested small RNN and Transformer networks on a limited number of tasks. Our work expands the injection method to much larger Transformer models. Furthermore, we focus on specific applications in the MNMT domain in regards to zero-shot tasks.

Other works have examined the impact of various architectural representations to increase model capacity and capability. Zhang et al. (2020) improved zero-shot translation by addressing off-target translation through random online back-translation. Other approaches include language dependent positional embeddings and hidden units (Wang et al., 2018) and dedicated encoders/decoders for each source and target language (Firat et al., 2016). Our work, in contrast, simply augments the MNMT system with additional information, while still maintaining the overall shared architecture across all languages. To our knowledge, this is the first study that investigates the concatenation of a language tag to the feed forward layers, within the realm of machine translation.

3 Methodology

We propose a novel method, *injection*, for distributing language tag information throughout the entire MNMT architecture, as opposed to being prepended to the encoder and/or decoder input alone. The injection method was first explored in regards to general controllable language gener-

Strategy	Source sentence	Target sentence	Encoder Injection	Decoder Injection
Existing Methods (Prompt tags only)				
T-∅/∅-∅	**<TGT>** Hello	Hola	None	None
T-T/∅-∅	**<TGT>** Hello	**<TGT>** Hola	None	None
∅-T/∅-∅	Hello	**<TGT>** Hola	None	None
ST-T/∅-∅	**<SRC>** **<TGT>** Hello	**<TGT>** Hola	None	None
Injection Methods (Ours)				
∅-∅/T-T	Hello	Hola	**<TGT>**	**<TGT>**
T-T/T-T	**<TGT>** Hello	**<TGT>** Hola	**<TGT>**	**<TGT>**
∅-T/∅-T	Hello	**<TGT>** Hola	None	**<TGT>**
∅-∅/S-T	Hello	Hola	**<SRC>**	**<TGT>**
ST-T/S-T	**<SRC>** **<TGT>** Hello	**<TGT>** Hola	**<SRC>**	**<TGT>**
∅-∅/ST-T	Hello	Hola	**<SRC>** + **<TGT>**	**<TGT>**

Table 1: Strategies tested, with and without our injection method. We label strategies with the format [Encoder text tag]-[Decoder text tag]/[Encoder injected tag]-[Decoder injected tag]. S indicates the language source tag (**<SRC>**) and T indicates the language target tag (**<TGT>**). ∅ indicates no tag input. The ∅-∅/ST-T strategy adds together the source and target tag embeddings for injection in the encoder.

ation tasks (Orten and Fulda, 2025). To test this method, we train 10 models, each using a different tagging strategy, both within prompts and with injection. We utilize the common encoder-decoder MNMT approach (Ha et al., 2016) with Transformers (Vaswani et al., 2017).

3.1 Language Tag Injection

We define a language tag as a unique token representing a language direction (source or target), e.g., '$< es >$' for indicating Spanish. In typical language tagging strategies, language tags are prefixed to the encoder and/or decoder inputs, thus learned by the language model implicitly.

In the injection method, the corresponding token for a language tag is embedded into an n-dimensional vector via the same learned embedding layer used in the encoder and decoder. This vector is then concatenated to the input of both linear layers in the feed-forward section of any encoder/decoder blocks, as can be seen in Figure 1. Thus, we are directly augmenting each point in the linear layers with tag information. Where t is the language tag embedding, W_i the linear layer weights, and x the input:

$$\text{FFN}(x) = (\max(0, (x \oplus t)W_1) \oplus t)W_2 \quad (1)$$

To accommodate the concatenation, we adjust the input size of the first linear layer to be $embedding_dim * 2$ and the input size of the second linear layer to be $ffn_dim + embedding_dim$

We test a variety of different approaches to including the language tag, both with and without injection. Throughout this work, we refer to each of our strategies by a code such as ([Encoder text tag]-[Decoder text tag]/[Encoder injected tag]-[Decoder injected tag]), using ∅ to represent no tag, S for a source language tag, and T for a target language tag.

A summary of all strategies tested can be found in Table 1. In general, we test four different approaches:

1. We test only including the textual target tag in the encoder (T-∅/∅-∅), following the suggestion of Wu et al. (2021).

2. We test including the target tag in the encoder and decoder, both without (T-T/∅-∅) and with (T-T/T-T) our injection strategy. We also test injection without including the tag in the prompt (∅-∅/T-T). The inclusion of the textual target tag as the first token passed through the decoder follows the work of Wang et al. (2018).

3. We test only including the textual target tag in the decoder, both without (∅-T/∅-∅) and with (∅-T/∅-T) our injection strategy.

4. We test including the textual source and target

tags in the encoder, and the textual target tag in the decoder, both without (ST-T/∅-∅) and with (ST-T/S-T) our injection strategy. This follows the approach made by N ElNokrashy et al. (2022). We also test this method of injection without the tags in the prompt (∅-∅/S-T), as well as adding the source and target tag embeddings together when performing injection in the encoder (∅-∅/ST-T).

3.2 Datasets

For all experiments, we use parallel text data from the Massively Multi-way-aligned Multilingual Corpus (MMMC) [1] in 22 different languages paired with English. We use a subset of the 98 languages in this dataset, including Arabic, Bulgarian, Chinese (Traditional), Czech, Dutch, French, German, Greek, Hungarian, Italian, Japanese, Persian, Polish, Portuguese, Romanian, Russian, Slovak, Slovenian, Spanish, Thai, Turkish, and Vietnamese. The total number of parallel sentences for both English-X and X-English directions is 37,299,606 sentence pairs. The number of parallel English-X sentences for each language are listed in Table 7 of Appendix A.

The MMMC dataset is comprised of parallel text translations derived from the translation memories of publicly available content provided on the website of The Church of Jesus Christ of Latter-day Saints [2]. This data contains translated sentences from various religious domains, including scripture, teachings, sermons, speeches, humanitarian resources, and administrative documents. All translations in the dataset were reviewed by professionally employed translators for quality and accuracy. We split our data into train, validation, and test sets. In order to evaluate zero-shot translation, we create a test and validation set which included translations common across all 23 languages. We choose to sample 500 validation and 1000 test sentences across 506 language directions (44 of which are English-centric). We train on the 37,233,606 remaining English-centric sentences which do not include any non-English X-Y paired data. We consider all X-Y pairs which do not include English to be zero-shot pairs.

[1] We have permission to use this data, though it has not yet been publicly released. Public release of this dataset is forthcoming.

[2] https://churchofjesuschrist.org

Hyperparameters	
FFN Dimension	4096 (2400 w/ injection)
Embedding Dimension	1024
Attention Heads	16
Layers	6
Sequence Length	512
Batch Size	1024
Learning Rate	0.0001
# Parameters	374M

Table 2: General hyperparameters used for all models in primary experiments.

3.3 Experimental Setup

We train all models from a random initialization. For architecture, we use the open-source version of BART (Lewis et al., 2020), available via HuggingFace. We modify the model code where necessary to enable injection, or concatenation of the embedded language tag, in the input to each feed-forward layer in every encoder and decoder block, as discussed in Section 3.1.

We generally follow the parameter guidelines of Transformer Big (Vaswani et al., 2017). Model parameters are summarized in Table 2. The feed-forward dimension sizes for all models using injection are adjusted to account for the additional parameters resulting from the injection method. This is done by decreasing the feed-forward network dimension to 2400. In this manner, all models have a parameter count within 1 million of 374M. We use a vocabulary size of 192,000, with a SentencePiece tokenizer (Kudo and Richardson, 2018).

All models are trained on 4 NVIDIA A100 GPUs. We use the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0001. We train until convergence, with a batch size of 1024 sentence pairs. Training to convergence took about 15 hours, on average. The best model checkpoints are then used for evaluation. We evaluate using BLEU (Papineni et al., 2002) and chrF (Popović, 2015) via the SacreBLEU implementation (Post, 2018), a standard evaluation suite for MNMT models.

4 Results

4.1 Performance across strategies

For every baseline strategy tested, there exists an equivalent method of language tag injection that

Strategy	BLEU		chrF	
	Supervised	Zero-Shot	Supervised	Zero-Shot
Existing Methods (Prompt tags only)				
T-∅/∅-∅	44.21 ± 2.94	21.38 ± 0.61	63.96 ± 2.66	44.87 ± 0.80
T-T/∅-∅	50.37 ± 2.58	29.45 ± 0.45	**67.85** ± 2.06	51.75 ± 0.49
∅-T/∅-∅	47.33 ± 2.46	28.87 ± 0.47	65.94 ± 2.01	51.05 ± 0.50
ST-T/∅-∅	**50.43** ± 2.56	29.47 ± 0.52	**67.85** ± 2.06	51.31 ± 0.50
Injection Methods (Ours)				
∅-∅/T-T	44.04 ± 2.91	22.46 ± 0.52	63.69 ± 2.64	46.59 ± 0.72
T-T/T-T	50.06 ± 2.52	29.84 ± 0.46	67.56 ± 2.04	51.96 ± 0.50
∅-T/∅-T	47.38 ± 2.46	29.62 ± 0.48	66.01 ± 1.98	52.02 ± 0.51
∅-∅/S-T	44.95 ± 2.85	24.97 ± 0.55	64.48 ± 2.57	48.04 ± 0.71
ST-T/S-T	50.19 ± 2.56	**30.77** ± 0.50	67.71 ± 2.07	**52.63** ± 0.50
∅-∅/ST-T	44.85 ± 2.86	24.80 ± 0.55	64.31 ± 2.58	47.87 ± 0.70

Table 3: Mean BLEU and chrF scores show improvement for zero-shot pairs with injection. Scores include margins representing 95% confidence intervals calculated from bootstrap resampling with 100,000 iterations. Margins for supervised pairs are notably large because of small sample size (44 supervised pairs).

yields higher performance on zero-shot tasks. As shown in Table 3, the mean BLEU and chrF scores for any method of tagging without injection is generally superior for supervised pairs, with ST-T/∅-∅ performing the best. However, mean scores for some equivalent injection strategies are higher on zero-shot pairs; namely, T-T/T-T, ∅-T/∅-T and ST-T/S-T. Notably, the strategies where only injection is done, without any tag in the prompt, do not perform as well. This suggests that the presence of the language tag within the prompt remains an important element of model conditioning.

Of particular interest in this work is not just the mean overall performance, but the improvements seen for specific language pairs. BLEU scores for individual pairs compared between equivalent strategies with and without injection are shown in Figures 2, 3, and 4. In each of these figures, points above the dotted red-line signify pairs that performed better, on average, with our injection models, compared to the respective baseline method. In Figure 3, we note a significant cluster of improved scores with the ∅-T/∅-T strategy, when compared to the ∅-T/∅-∅ strategy.

Overall, the best tagging method for zero-shot translation is ST-T/(S-T), shown in Figure 4. This matches the suggestion made by N ElNokrashy et al. (2022), with the inclusion of injection. An even more exaggerated cluster of improved scores appears, all pairs with Thai as the target language.

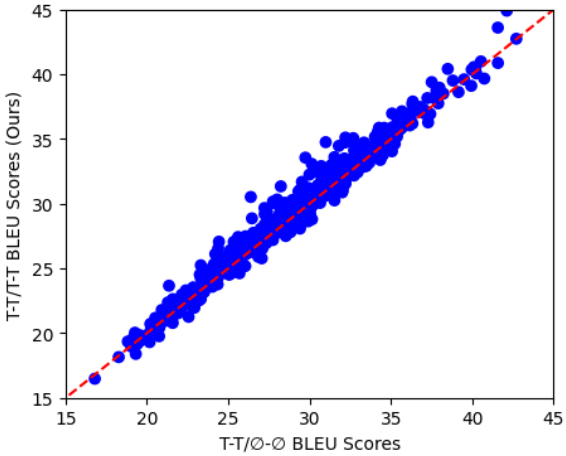

Figure 2: BLEU score for all language pairs between the prompt-only (T-T/∅-∅) and our injection (T-T/T-T) method. Improvement from injection in this case appears minimal.

We investigate this phenomenon further in Section 4.2.

To further explore what benefits tag injection brings to specific pairs, we show the mean zero-shot BLEU score improvement for language directions when injection is added. In Table 4, we observe that the addition of tag injection improves BLEU scores by up to 1-2 points for certain language pairs. Most notably, pairs with Thai as the target language experience an improvement of 4-6 points, which we explore in Section 4.2. We

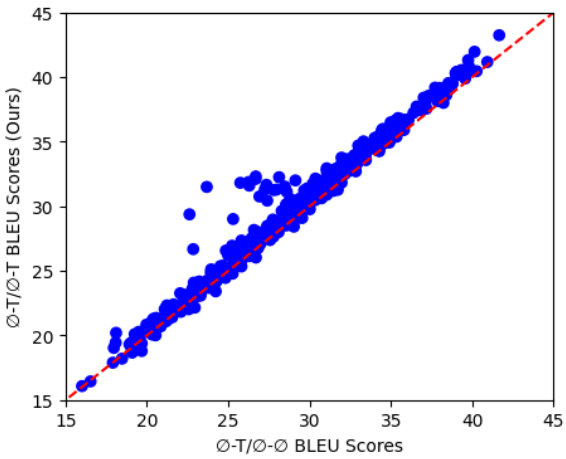

Figure 3: Our decoder-only injection (∅-T/∅-T) provides improvement for certain language pairs over the prompt-only strategy (∅-T/∅-∅).

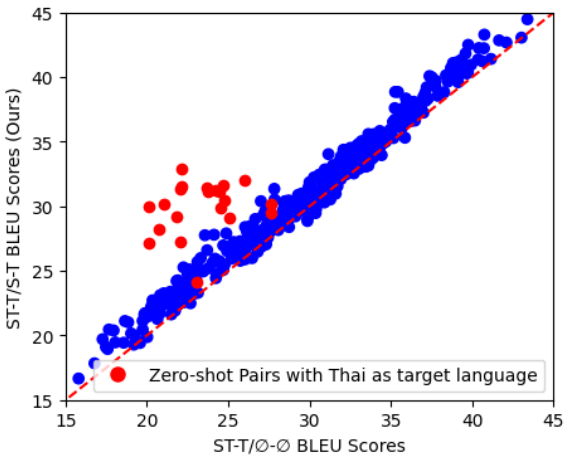

Figure 4: When injection is used in this instance, we note a significant improvement on a cluster of pairs where Thai is the target language. Our injection method also provides at least a marginal improvement for almost all pairs. Red points signify zero-shot pairs with Thai as the target language.

did not find any correlation between language resource level and performance, suggesting that, in this data, the injection method does not improve low-resource pairs.

4.2 Removing Thai

To further investigate the perceived dramatic improvements with Thai language pairs, we (1) train a model with the ST-T/(S-T) strategy again with a different seed, to ensure the consistency of the results regardless of initialization, and (2) train equivalent models without the Thai language pairs. Training with a different seed yielded comparable results, with the injection model still learning significantly better on Thai target language pairs, when compared to the baseline method. Figure 5 shows that removing the Thai language pairs yields an injection model without any specific language pair cluster such as before.

Upon further investigation into our dataset, we found evidence that some Thai target pairs contain instances of English phrases and titles not present in other target pairs. Even if these pairs caused the observed Thai improvements, it remains that only the injection models benefited from them. We hypothesize that the injection method may have been able to take greater advantage of the anomalies present in the data. It is also possible that injection may allow the model to generalize its knowledge more fully when translating into the Thai writing system, a script that is not heavily represented in the overall corpus. We leave further investigation to future work.

4.3 Varying model dimensions

In our core experiments, we adjusted the feed-forward dimensions of the models using injection, in order to account for the additional parameters resulting from injection. In general, this meant that the baseline models were trained with a feed-forward layer dimension of 4096 in both encoder and decoder, while the injection models use a feed-forward layer dimension of 2400. We posit that this approach makes the most sense; the injection method only impacts the feed-forward layers in the model, so by lowering the feed-forward dimension we are adjusting the model parameters closest to the injection.

To ensure that varying this dimension did not impact the performance of the models in other unexpected ways, we train several models with different model dimension adjustments. Comprehensive details on parameter adjustments can be found in Table 5. For all of these tests, we use the T-T/(T-T) tagging method.

We use model V1 as the baseline against 3 variations of the injection method: smaller FFN dimension (V2), fewer layers (V3), and a smaller embedding dimension (V4). Results can be seen in Table 6. We observe that V2 and V4 achieve superior performance over the baseline on zero-shot pairs, with V2 being the best, while V3 is marginally worse. This confirms our belief that adjusting the feed-forward dimension makes the most sense for the injection approach.

Strategy	Language Direction	Zero-shot Improvement
T-T/(T-T)	Slovak→X	+1.93
	Slovenian→X	+1.70
	Czech→X	+1.66
	X→French	+1.59
	X→Italian	+0.88
	...	
	Dutch→X	-0.22
	German→X	-0.42
	Japanese→X	-0.44
∅-T/(∅-T)	X→Thai	+4.20
	X→Slovak	+1.27
	X→Czech	+1.18
	French→X	+1.15
	Spanish→X	+1.08
	...	
	X→Persian	+0.17
	X→German	+0.11
	X→Turkish	-0.10
ST-T/(S-T)	X→Thai	+6.56
	X→Italian	+2.25
	X→Turkish	+2.05
	Chinese→X	+1.95
	Spanish→X	+1.94
	...	
	X→Arabic	+0.67
	X→Polish	+0.31
	X→Slovenian	-0.20

Table 4: Language pairs with highest BLEU score point improvement using our injection method, over the equivalent baseline strategy without injection. We also show directions with the least improvement.

Finally, we train an injection model with the default parameters (V5), and adjust a model without injection up to the number of parameters of the first model (V6). This method could be seen as the "other side of the coin"; rather than adjusting the injection model parameters down, we adjust the default model parameters up. Interestingly, we observe worse performance with the injection model. We hypothesize that the additional parameters from the injection approach act supplementary to the model, rather than primary. By adjusting the baseline model parameters up, we are effectively giving the baseline model more primary parameter space to adjust.

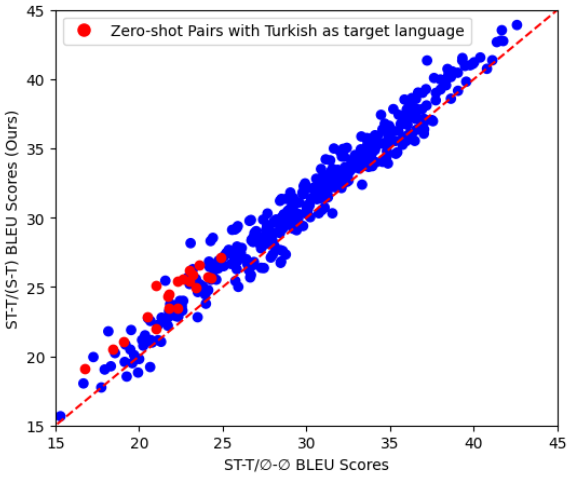

Figure 5: Scores after removing Thai language pairs. While no specific cluster as dramatic as Thai exists, there are still concentrated clusters for languages such as Turkish. Red points signify zero-shot pairs with Turkish as the target language.

We emphasize that many of the benefits from tag injection occur with individual language pairs, which the mean scores for BLEU or chrF do not fully represent. However, the mean allows us to interpret general performance.

4.4 Varying Layer Injection

To discover the impact of tag injection across layers, we train models using the T-T/(T-T) strategy and vary the number of layers that injection is performed on in the encoder and decoder. We find that performance increases as more layers use injection, suggesting that injection acts more like noise when it is not fully distributed across the system. This behavior is fairly intuitive, and it confirms our belief that injection contributes information to the model. Furthermore, we find that injection impacts overall performance more dramatically when used in the encoder than when used in the decoder, as seen in Figure 6.

5 Conclusion

In this work, we proposed a novel method for language tagging, accomplished by concatenating the embedded vector of a language tag to the input of linear layers throughout an encoder/decoder model. We refer to this approach as tag injection. We explored this method in relation to a variety of previously proposed language tagging strategies and tested on a dataset that will be released publicly.

Our results show that tag injection may provide

Hyperparameters	V1	V2	V3	V4	V5	V6
Injection	No	Yes	Yes	Yes	Yes	No
Embedding Dim	1024	1024	1024	896	1024	1024
FFN Dim	4096	2400	4796	4096	4096	6656
Heads	16	16	16	16	16	16
Layers	6	6	4	6	6	6
# Parameters	374M				436M	

Table 5: We test variations of model dimensions, while still matching the same parameter size.

Test	BLEU		chrF	
	Supervised	**Zero-Shot**	**Supervised**	**Zero-Shot**
V1	**50.37**	29.45	**67.85**	51.75
V2	50.06	**29.84**	67.56	**51.96**
V3	49.40	29.00	67.05	51.08
V4	49.76	29.71	67.33	51.75
V5	50.32	30.58	67.67	52.30
V6	**50.53**	**32.09**	**67.87**	**53.58**

Table 6: Adjusting the FFN dimension (V2) and the embedding dimension (V4) both show improvement over the baseline (V1) for zero-shot pairs. Adjusting the FFN dimension of the model without injection (V6) to match the number of parameters of the model with injection (V5) yields a non-injection model with higher performance all around.

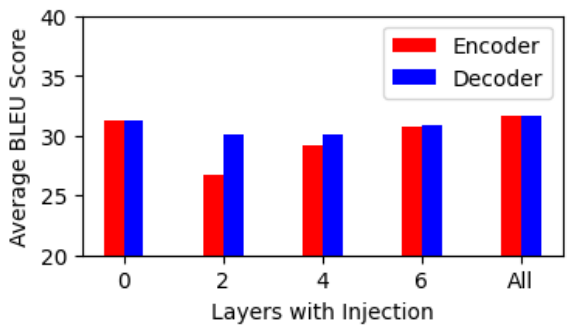

Figure 6: Injection may act like noise until it is fully distributed throughout the model. For the encoder experiments, no injection was performed in the decoder, and vice versa with the decoder experiments.

a performance benefit, in terms of BLEU and chrF scores, to certain zero-shot language pairs across multiple tagging strategies. Furthermore, we confirm the conclusion made by N ElNokrashy et al. (2022) that inputting the source and target tag in the encoder, and the target tag in decoder, is a very effective tagging strategy. We explored the robustness of the injection method by varying model dimensions and layers with injection, finding that the method provides meaningful information to the model, rather than simply acting as noise.

Tag injection only requires a relatively simple modification to any encoder/decoder architecture; as such, this tagging method could be applied across a wide range of MNMT systems, particularly those that focus on many zero-shot directions. We posit that the injection method, and language tagging in general, remains relevant within the rapidly changing landscape of MNMT because it provides explicit conditioning to a translation model, an element that becomes critical for smaller models designed for specific tasks. Future work in this area includes the application of language tag injection to other machine translation tasks, specifically those focused on low-resource and zero-shot challenges, as well as further exploration into the learning behavior of models with injection on specific language directions.

Limitations

We used a single dataset containing translation pairs for a specific domain. Future work should include the extension of the injection method to other datasets and domains, including variations on supervised and zero-shot pair composition. We

also acknowledge that we primarily rely on BLEU and chrF scores for evaluation, and future work should apply other metrics in order to gain a more holistic idea of performance when using injection.

We acknowledge that this work focuses on medium-scale Transformer models for machine translation, and, by consequence, is not directly comparable to the latest large-scale multi-language pre-trained models. The focus of these experiments was to conduct low-cost investigation across a broad range of techniques, and future work should apply the best approaches towards large-scale experiments.

6 Ethics Statement

Data from the MMMC corpus is derived from publicly available information from The Church of Jesus Christ of Latter-day Saints website. The corpus contains scriptures, doctrines, and teachings of The Church of Jesus Christ of Latter-day Saints, with which people of differing faiths and belief systems may disagree. Some names of individuals and other limited information about them (but not what is normally considered personally identifiable information, or PII) are included in the corpus, though the information is publicly available on the Church's website, as stated above.

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.

Graeme Blackwood, Miguel Ballesteros, and Todd Ward. 2018. Multilingual neural machine translation with task-specific attention. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3112–3122, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

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. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. 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.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. 2019. Improved zero-shot neural machine translation via ignoring spurious correlations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1258–1268, Florence, Italy. Association for Computational Linguistics.

Thanh-Le Ha, Jan Niehues, and Alex Waibel. 2016. Toward multilingual neural machine translation with universal encoder and decoder. In *Proceedings of the 13th International Conference on Spoken Language Translation*, Seattle, Washington D.C. International Workshop on Spoken Language Translation.

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 Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Taku Kudo and John Richardson. 2018. 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. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Muhammad N ElNokrashy, Amr Hendy, Mohamed Maher, Mohamed Afify, and Hany Hassan. 2022. Language tokens: Simply improving zero-shot multi-aligned translation in encoder-decoder models. In *Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 70–82, Orlando, USA. Association for Machine Translation in the Americas.

Jay Orten and Nancy Fulda. 2025. Improving controlled text generation via neuron-level control codes. In

Proceedings of the 17th International Conference on Agents and Artificial Intelligence - Volume 3: ICAART, pages 574–581. INSTICC, SciTePress.

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.

Ngoc-Quan Pham, Jan Niehues, Thanh-Le Ha, and Alexander Waibel. 2019. Improving zero-shot translation with language-independent constraints. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 13–23, Florence, Italy. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yining Wang, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2018. Three strategies to improve one-to-many multilingual translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2955–2960, Brussels, Belgium. Association for Computational Linguistics.

Rachel Wicks and Kevin Duh. 2022. The effects of language token prefixing for multilingual machine translation. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 148–153, Online only. Association for Computational Linguistics.

Liwei Wu, Shanbo Cheng, Mingxuan Wang, and Lei Li. 2021. Language tags matter for zero-shot neural machine translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3001–3007, Online. Association for Computational Linguistics.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

A Dataset

Language	Number of Pairs
Arabic	88,243
Bulgarian	540,396
Chinese	536,251
Czech	588,943
Dutch	838,413
French	1,849,045
German	1,466,305
Greek	258,307
Hungarian	751,229
Italian	1,714,727
Japanese	1,343,870
Persian	52,322
Polish	620,554
Portuguese	2,105,240
Romanian	641,284
Russian	1,325,292
Slovak	181,270
Slovenian	177,493
Spanish	2,272,917
Thai	726,979
Turkish	68,566
Vietnamese	501,057

Table 7: Counts of the number of sentence pairs with English for each of the 22 languages in our dataset.