The MLLP-UPV Spanish-Portuguese and Portuguese-Spanish Machine Translation Systems for WMT19 Similar Language Translation Task

Pau Baquero-Arnal, Javier Iranzo-Sánchez, Jorge Civera, Alfons Juan
Machine Learning and Language Processing (MLLP) research group
Valencian Research Institute for Artificial Intelligence (VRAIN)
Universitat Politècnica de València
Camí de Vera s/n, 46022, València, Spain
{pabaar,jairsan,jcivera,ajuan}@vrain.upv.es

Abstract

This paper describes the participation of the MLLP research group of the Universitat Politècnica de València in the WMT 2019 Similar Language Translation Shared Task. We have submitted systems for the Portuguese ↔ Spanish language pair, in both directions. They are based on the Transformer architecture as well as on a novel architecture called 2D alternating RNN. Both systems have been domain adapted through fine-tuning that has been shown to be very effective.

1 Introduction

In this paper we describe the supervised Statistical Machine Translation (MT) systems developed by the MLLP research group of the Universitat Politècnica de València for the Related Languages Translation Shared Task of the *ACL 2019 Fourth Conference on Machine Translation* (WMT19). For this task, we participated in both directions of the Portuguese ↔ Spanish language pair using Neural Machine Translation (NMT) models. This paper introduces a novel approach to translation modeling that is currently being developed. We report results for this approach and compare them with models based on the well-performing Transformer (Vaswani et al., 2017) NMT architecture. A domain adapted version of this latter system achieves the best results out of all submitted systems on both directions of the shared task.

The paper is organized as follows. Section 2 describes the architecture and settings of the novel 2D RNN model. Section 3 describes our baseline systems and the results obtained. Section 4 reports the results obtained by means of the fine-tuning technique. Section 5 reports comparative results with respect to the systems submitted by the other competition participants. Section 6 outlines our conclusions for this shared task.

2 2D Alternating RNN

In this section, we will describe the general architecture of the 2D alternating RNN model. The 2D alternating RNN is a novel translation architecture in development by the MLLP group. This architecture approaches the machine translation problem with a two-dimensional view, much in the same manner as Kalchbrenner et al. (2015); Bahar et al. (2018) and Elbayad et al. (2018). This view is based on the premise that translation is fundamentally a two-dimensional problem, where each word of the target sentence can be explained in some way by all the words in the source sentence. Two-dimensional translation models define the distribution $p(e_i|f_0^J, e_0^{i-1})$ by jointly encoding the source sentence (f_0^J) and the target history (e_0^{i-1}), whereas the usual translation models encode them separately, in separate components usually called "encoder" and "decoder".

The proposed architecture is depicted in Figure 1. It defines a two-dimensional translation model by leveraging already known recurrent cells, such as LSTMs or GRU, without any further modification.

As many other translation models, we have a context vector which is projected to vocabulary size and a softmax (σ) is applied to obtain the probability distribution of the next word at timestep i:

$$p(e_i = x|f_0^J, e_0^{i-1}) = \sigma(W c_i)_x \qquad (1)$$

To explain how this context vector is drawn from a two-dimensional processing style, we need to define a grid with two dimensions: one for the source, and one for the target. From this point, we will define a layer-like structure called block, where each block of the model has such a grid as the input, and another one as the output.

Proceedings of the Fourth Conference on Machine Translation (WMT), Volume 3: Shared Task Papers (Day 2) pages 179–184

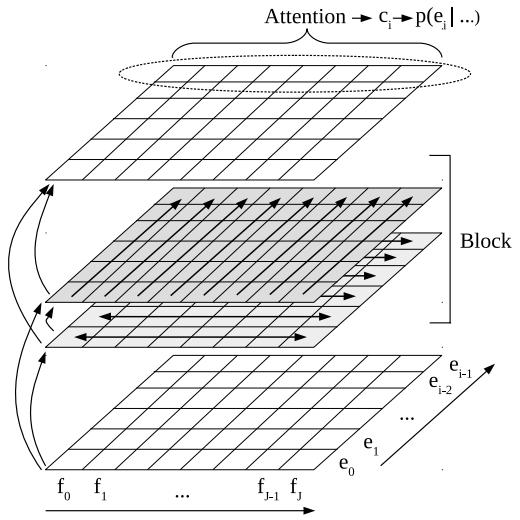

Figure 1: The 2D alternating RNN architecture. White grids on the top and bottom represent the input/output of a block. Arrows in grey grids represent the RNNs, while the arrows on the left depict how the layers are interconnected. Arrows on the bottom and bottom right indicate the source and target dimensions.

The first grid that serves as input to this two-dimensional architecture has each cell s^0_{ij} containing the concatenation of the source embedding in position j and the target embedding in position $i-1$:

$$s^0_{ij} = \begin{bmatrix} f_j \\ e_{i-1} \end{bmatrix} \quad (2)$$

Each block of the model has two recurrent cells: one along the source dimension and another one along the target dimension. They process each row or column independently of one another. The horizontal cell is bidirectional and receives the grid s^l as its input:

$$h^l_{ij} = \begin{bmatrix} \mathrm{RNN_{h1}}(h^l_{i,j-1}, s^{l-1}_{ij}) \\ \mathrm{RNN_{h2}}(h^l_{i,j+1}, s^{l-1}_{ij}) \end{bmatrix} \quad (3)$$

The vertical cell receives the concatenation of h^l and s^l:

$$k^l_{ij} = \mathrm{RNN_k}(k^l_{i-1,j}, [s^{l-1}_{ij}; h^l_{ij}]) \quad (4)$$

The output of the block is the concatenation of the output of both cells:

$$s^l_{ij} = \begin{bmatrix} h^l_{ij} \\ k^l_{ij} \end{bmatrix} \quad (5)$$

From the output of the last block, s^L, we generate a context vector as follows:

$$c_i = \mathrm{Attention}([\mathrm{s^L_{i0}}, \ldots, \mathrm{s^L_{iJ}}]) \quad (6)$$

The Attention function extracts a single vector from a set of vectors leveraging an attention mechanism. That is, it scores the vectors according to a learned linear scoring function, which is followed by a softmax to extract scores; and with those scores it performs a weighted sum to obtain a context vector.

3 Baseline systems

This section describes training corpora as well as the baseline model architectures and configurations adopted to train our NMT systems. As said in Section 1, two different model architectures were trained: the Transformer architecture (Vaswani et al., 2017) and our proposed 2D alternating RNN architecture. BLEU (Papineni et al., 2002) scores were computed with the `multi-bleu` utility from Moses (Koehn et al., 2007).

3.1 Corpus description and data preparation

The training data is made up of the JCR, Europarl, news-commentary and wikititles corpora. Table 1 shows the number of sentences, number of words and vocabulary size of each corpus. The provided development data was split equally in two disjoint sets, and one was used as development set and the other as test set.

Corpus	Sent.(K)	Words(M)		Vocab.(K)	
		Es	Pt	Es	Pt
JCR	1650	42	40	264	264
Europarl	1812	53	52	177	156
news	48	1	1	49	47
wikititles	621	1	1	292	295
dev	1.5	0	0	6	6
test	1.5	0	0	6	6
Total	4131	98	96	623	604

Table 1: Statistics of the data sets used to train the Spanish ↔ Portuguese MT systems.

The data was processed using the standard Moses pipeline (Koehn et al., 2007), specifically, punctuation normalization, tokenization and truecasing. Then, we applied 32K BPE (Sennrich et al., 2016b) operations, learned jointly over the source and target languages. We included in the

vocabulary only those tokens occurring at least 10 times in the training data.

3.2 Transformer baseline models

For the Transformer (Vaswani et al., 2017) models, we used the "Base" configuration (512 model size, 2048 feed-forward size), trained on one GPU. The batch size was 4000 tokens, and we carried out gradient accumulation by temporarily storing gradients and updating the weights every 4 batches. This setup allowed us to train models using an effective batch size of 16000 tokens. We used dropout (Srivastava et al., 2014) with 0.1 probability of dropping, and label smoothing (Szegedy et al., 2016) where we distribute 0.1 of the probability among the target vocabulary. We stored a checkpoint every 10000 updates, and for inference we used the average of the last 8 checkpoints.

We used the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.98$. The learning rate was updated following an inverse square-root schedule, with an initial learning rate of $5 \cdot 10^{-4}$ and 4000 warm-up updates.

The models were built using the fairseq toolkit (Ott et al., 2019).

3.3 2D alternating RNN baseline model

For the 2D alternating RNN models, we used GRU as the recurrent cell, 256 for the embedding size and 128 as the number of units of each layer of the block. The model consisted of a single block. The batch size was 20 sentences, with a maximum length of 75 subword units.

We used the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$. The learning rate was initialized at 10^{-3} and kept constant, but halved after 3 checkpoints without improving the development perplexity. A checkpoint was saved every 5000 updates. The model was built using our own toolkit. Due to time constraints, the 2D alternating model was only trained for the Portuguese → Spanish direction.

3.4 Results

Table 2 shows the evaluation results for the Portuguese→Spanish systems, and Table 3 shows the evaluation results for our Spanish→Portuguese Transformer system. For the Portuguese → Spanish direction, the Transformer model obtains 57.4 BLEU in the test set, and 51.9 in the hidden test set of the competition.

	BLEU	
System	test	test-hidden
Transformer	57.4	51.9
2D altern. RNN	55.1	49.7

Table 2: Baseline BLEU scores on the Portuguese → Spanish task.

	BLEU	
System	test	test-hidden
Transformer	51.2	45.5

Table 3: Baseline BLEU scores on the Spanish → Portuguese task.

The 2D alternating model achieves 55.1 and 49.7 BLEU, respectively. These results show how, even though it is in early stages of development, the 2D alternating RNN model is able to obtain competitive results for this task that are not very far from those obtained by the state-of-the-art Transformer architectures. It is worth noting that this has been achieved with a model that has significantly fewer parameters (14.9M) than its Transformer counterpart (60.2M).

4 Fine-tuning

NMT models perform best when trained with data from the domain of the test data. However, most available parallel corpora belong to institutional documents or internet-crawled content domains, so it is common to find situations where there is a domain mismatch between train and test data. In such cases, small amounts of in-domain data can be used to improve system performance by carrying out an additional training step, often referred to as the fine-tuning step, using the in-domain data after the main training finishes. This technique has been used to adapt models trained with general domain corpora to specific domains with only small amounts of in-domain data (Luong and Manning, 2015; Sennrich et al., 2016a).

In order to empirically test if this is one of such cases, we have trained two language models, one using only the presumably out-of-domain data (the train corpora from Table 1), and one using only the in-domain development data. The models were 4-gram language models trained using the SRI Language Modelling Toolkit (Stolcke et al., 2011). We then computed the perplexity of the test set using these two language models. The model that was trained with the out-of-domain data obtains a per-

System	BLEU test	BLEU test-hidden
Transformer	57.4	51.9
+ fine-tuned	72.4	66.6
2D altern. RNN	55.1	49.7
+ fine-tuned	64.0	-

Table 4: Comparative BLEU scores of the Transformer and 2D alternating RNN models on the Portuguese → Spanish task.

System	BLEU test	BLEU test-hidden
Transformer	51.3	45.5
+ fine-tuned	70.7	64.7

Table 5: Comparative BLEU scores of the Transformer model on the Spanish → Portuguese task.

plexity of 298.0, whereas the model that used the in-domain data obtains a perplexity of 81.9. This result shows that there is in a fact a domain mismatch between the train and test data, which supports the idea of carrying out fine-tuning.

We applied this to both translation directions, using the first part of the development data as in-domain training data, and the second part as a new dev set. One checkpoint was stored after every fine-tuning epoch, and we monitored model performance on the new dev set in order to stop fine-tuning once the BLEU results started decreasing. For the Transformer models, we used the same learning rate as when training stopped, while for the 2D alternating models we used 10^{-3}.

Tables 4 and 5 compare the BLEU scores achieved by the fine-tuned systems with that of the baseline non fine-tuned ones on the Portuguese→Spanish and Spanish→Portuguese tasks, respectively.

Table 4 shows that for this particular task, fine-tuning is a key step for achieving very substantial performance gains: in the Portuguese→Spanish task, we obtained a 15.0 BLEU improvement in the test set and a 14.7 BLEU improvement in the hidden test set for the Transformer model. The 2D alternating RNN obtained a 8.9 BLEU improvement thanks to fine-tuning. This also applies to the Spanish→Portuguese task, shown in Table 5: we obtained a 19.4 BLEU improvement in the test set, and a 19.2 BLEU improvement in the hidden test set after applying fine-tuning.

In order to understand the impact and behaviour

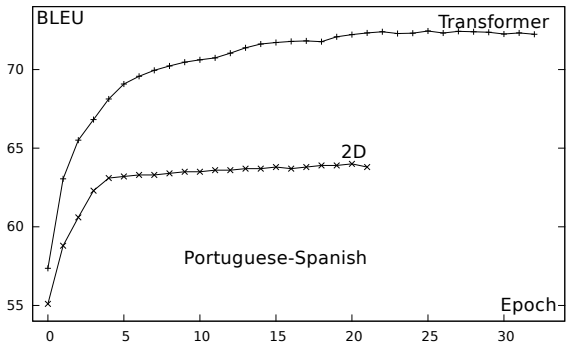

Figure 2: BLEU scores as a function of the number of fine-tuning epochs on the Transformer and 2D alternating RNN models for the Portuguese→Spanish task.

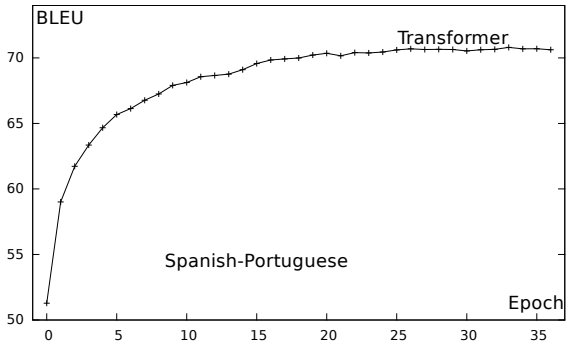

Figure 3: BLEU scores as a function of the number of fine-tuning epochs on the Transformer model for the Spanish→Portuguese task.

of the fine-tuning process, we have analyzed the model's performance as a function of the number of fine-tuning epochs. Figure 2 shows the impact of the fine-tuning step for the Transformer and 2D alternating RNN models on the Portuguese → Spanish task, while Figure 3 shows the results of the fine-tuning step applied to the Transformer model on the Spanish → Portuguese task. In both language pairs, the first epochs are the most beneficial for system performance, and additional fine-tuning epochs bring diminishing returns until the BLEU curve flattens.

5 Comparative results

We now move on to the results for the primary submissions of all participants in the Shared Task. We chose to send our fine-tuned Transfomer systems as primary submissions to both tasks after reviewing the results on the provided test set (Section 4). The submission was made with the checkpoint that achieved the best performance on the fine-tuning dev data. Table 6 shows the results

Team	BLEU	TER
MLLP	**66.6**	**19.7**
NICT	59.9	25.3
U. Helsinki	58.4	25.3
Kyoto U.	56.9	26.9
BSC	54.8	29.8
UBC-NLP	52.3	32.9

Table 6: Primary submission results of the Portuguese → Spanish shared task in the hidden test set.

Team	BLEU	TER
MLLP	**64.7**	**20.8**
UPC-TALP	62.1	23.0
NICT	53.3	29.1
U. Helsinki	52.0	29.4
UBC-NLP	46.1	36.0
BSC	44.0	37.5

Table 7: Primary submission results of the Spanish → Portuguese shared task in the hidden test set.

of the Portuguese→Spanish task, while Table 7 shows the results of the Spanish→Portuguese task; both in BLEU and TER (Snover et al., 2006).

In both tasks, our system outperformed all other participants by a significant margin. In the Portuguese→Spanish task, our submission outperforms the next best system by 6.7 BLEU and 5.6 TER. In a similar manner, our submission to the Spanish → Portuguese task improves the results of the second-best submission by 2.6 BLEU and 2.2 TER points. We attribute our success to the domain adaptation carried out by means of the fine-tuning technique. We have been able to apply this technique by using part of the competition's development data as in-domain training data.

6 Conclusions

We have taken on the similar language task with the same approaches that we found useful for other kinds of translation tasks. NMT models, specifically the Transformer architecture, fare well in this task without making any specific adaptation to the similar-language setting. In fact, we achieved the best results among the participants using a general domain-adaptation approach.

For this particular task, the use of in-domain data to carry out fine-tuning has allowed us to obtain remarkable results that significantly outperform the next best systems in both Portuguese→Spanish and Spanish→Portuguese. We believe these results are explained by the domain difference between training and test data, and are unrelated to the similarity between Spanish and Portuguese.

We have introduced the 2D alternating RNN model, a novel NMT architecture, that has been tested in the Portuguese→Spanish task. With small embedding and hidden unit sizes and a shallow architecture, we achieved similar performance to the Transformer model, although the difference between them increases after applying fine-tuning.

In terms of future work, we plan to fully develop the 2D alternating RNN model in order to support larger embedding and hidden unit sizes as well as deeper architectures using more regularization. All these improvements should allow us to increase the already good results achieved by this model.

Acknowledgments

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 761758 (X5gon); the Government of Spain's research project Multisub, ref. RTI2018-094879-B-I00 (MCIU/AEI/FEDER, EU) and the Generalitat Valenciana's predoctoral research scholarship ACIF/2017/055.

References

Parnia Bahar, Christopher Brix, and Hermann Ney. 2018. Towards two-dimensional sequence to sequence model in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3009–3015, Brussels, Belgium. Association for Computational Linguistics.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2018. Pervasive attention: 2D convolutional neural networks for sequence-to-sequence prediction. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 97–107, Brussels, Belgium. Association for Computational Linguistics.

Nal Kalchbrenner, Ivo Danihelka, and Alex Graves. 2015. Grid long short-term memory. *arXiv preprint*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference for Learning Representations*, San Diego, California, USA.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

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, July 6-12, 2002, Philadelphia, PA, USA.*, pages 311–318.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. SRILM at Sixteen: Update and Outlook. In *Proceedings of the 2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Waikoloa, Hawaii, USA.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6000–6010.