Learning to Solve NLP Tasks in an Incremental Number of Languages

Giuseppe Castellucci
Amazon
Seattle, USA
giusecas@amazon.com

Simone Filice
Amazon
Tel Aviv, Israel
filicesf@amazon.com

Danilo Croce
Dept. of Enterprise Engineering
University of Rome, Tor Vergata
Roma, Italy
croce@info.uniroma2.it

Roberto Basili
Dept. of Enterprise Engineering
University of Rome, Tor Vergata
Roma, Italy
basili@info.uniroma2.it

Abstract
In real scenarios, a multilingual model trained to solve NLP tasks on a set of languages can be required to support new languages over time. Unfortunately, the straightforward retraining on a dataset containing annotated examples for all the languages is both expensive and time-consuming, especially when the number of considered languages grows. Moreover, the original annotated material may no longer be available due to storage or business constraints. Re-training only with the new language data will inevitably result in Catastrophic Forgetting of previously acquired knowledge. We propose a Continual Learning strategy that updates a model to support new languages over time, while maintaining consistent results on previously learned languages. We define a Teacher-Student framework where the existing model “teaches” to a student model its knowledge about the languages it supports, while the student is also trained on a new language. We report an experimental evaluation in several tasks including Sentence Classification, Relational Learning and Sequence Labeling.

1 Introduction
In Natural Language Processing (NLP), multilingualism refers to the capability of a single model to cope with multiple languages. Recently, different Transformer-based architectures have been extended to operate over multiple languages, as in Conneau et al. (2020); Conneau and Lampl (2019); Pires et al. (2019). Despite these models can be applied in the zero-shot setting (Xian et al., 2019; Artetxe and Schwenk, 2019), in many practical applications their quality will not be satisfactory. Instead, fine-tuning over annotated material in each target language is needed to obtain competitive results, as the experimental results in Lewis et al. (2019); Tran and Bisazza (2019) suggest. Having annotated material for all the languages is not always possible, especially when the model has to support an incremental number of new languages over time. In fact, the original fine-tuning material may no longer be available for storage, business or privacy constraints. For example, in a real-world application, customers may request deletion of their data, or the service itself may provide specific data retention policies, or the adopted model may be provided by a third party that did not release the training data (Chen and Moschitti, 2019). In these cases, new language support can be added in a Continual Learning (CL) setting (Lange et al., 2019), that is fine-tuning the model only using the annotated material for the new language(s). However, this approach is vulnerable to the Catastrophic Forgetting (CF) (McCloskey and Cohen, 1989) of previously learned languages, a well-documented concern discussed in Chen et al. (2018): when a model is incrementally fine-tuned on new data distributions, it risks forgetting how to treat instances of the previously learned ones.

In this paper, we propose a CL strategy for updating a model over an incremental number of languages, so that at each step the model requires only annotated examples of the new language(s). Our goal is to remove the dependency on the original fine-tuning material and reduce the need for annotated data at each training step. We propose a Teacher-Student framework inspired by the Knowledge Distillation (KD) literature (Hinton et al., 2015). Although this technique is traditionally used for the purpose of model compression (Sanh et al., 2019), recent works in Computer Vision applied KD to incrementally learn image processing tasks (Li and Hoiem, 2018). Here, we adopt KD to miti-
gate CF when incrementally training Transformer-based architectures (Devlin et al., 2019) for semantic processing tasks. The existing model (here the teacher) imparts knowledge to a (student) model about the languages it already supports, while this is trained on new languages.

We evaluated our approach using multilingual BERT-based models on three semantic processing tasks, involving Sentence Classification, Paraphrase Identification and Sequence Tagging. Results suggest that the model can progressively learn new languages, while maintaining or even improving its quality over previously observed ones.

2 Related Work

Continual Learning (CL) (Chen et al., 2018) studies how to train a machine from a stream of data, which can evolve over time by changing the input distribution or by incorporating new tasks. CL aims to gradually extend the knowledge in a model (Lange et al., 2019), while avoiding Catastrophic Forgetting (Goodfellow et al., 2013). Previous work has mostly focused on Computer Vision (Shmelkov et al., 2017; Li and Hoiem, 2018; Rannen et al., 2017) by using Knowledge Distillation (KD) (Hinton et al., 2015) as the base framework.

CL in NLP, as opposed to Computer Vision, is still nascent (Greco et al., 2019; Sun et al., 2020). This reflects in the small number of proposed methods to alleviate CF, as discussed in Biesialska et al. (2020). In this context, some works focus on the Online Learning aspect of the CL (Filice et al., 2014). In NLP, KD has been mainly adopted to compress models (Kim and Rush, 2016; Sanh et al., 2019), and was only recently applied for CL in Named Entity Recognition (Monaikul et al., 2021).

In the context of multilingual analysis, most of the works leverage Domain Adaptation techniques within Machine Translation (Dong et al., 2015; Firat et al., 2016; Ha et al., 2016; Johnson et al., 2017; Tan et al., 2019) in order to apply a machine translation model to an increasing set of languages.

To the best of our knowledge, this is the first work adopting CL to mitigate CF when training Transformer-based models in an incremental number of languages for semantic processing tasks.

3 CL for Multilingual processing

Multilingual Continual Learning. In the targeted scenario, we have a multilingual neural model, namely $\mathcal{M}_{LA}$, originally pre-trained on a set of languages $L_P = \{l_1, l_2, \ldots\}$ (such as multilingual BERT (Pires et al., 2019)) and already fine-tuned to solve a task $T$ (such as sentence classification) on a given set of languages $L_A \subset L_P$. The scope is to extend such model to solve $T$ on a set of new languages $L_B \subset L_P$, with $L_A \cap L_B = \emptyset$.

In the rest of the discussion, without loss of generality, we assume that $L_B = \{l_{new}\}$, i.e., we support only one new language at a time. In case $n > 1$ new languages need to be added, a sequence of $n$ model extensions can be performed. In our setting, we assume that: (i) a new annotated dataset $S_{\{l_{new}\}}$ for task $T$ in language $l_{new}$ is available; (ii) the examples used to fine-tune $\mathcal{M}_{LA}$ are not available anymore; (iii) unlabeled examples are available in each language from $L_A$. Since $l_{new} \notin L_P$, i.e., the original pre-training stage included $l_{new}$, the model could already operate in a zero-shot setting (i.e., without any fine-tuning stage involving $l_{new}$ data). However, the performance of the zero-shot setting is typically non-satisfactory and a dedicated fine-tuning on $l_{new}$ is generally required. A naive CL strategy consists of fine-tuning $\mathcal{M}_{LA}$ over $S_{\{l_{new}\}}$. However, even though this schema is supposed to produce an effective model for $l_{new}$ instances, it is not guaranteed that the resulting model would still be competitive on languages $L_A$, due to CF (Greco et al., 2019; Sun et al., 2020). An alternative greedy solution consists of adopting self-training as in Rosenberg et al. (2005): $\mathcal{M}_{LA}$ is used to annotate some unlabeled examples in languages $L_A$ so that the resulting pseudo-labeled dataset $\hat{S}_{LA}$ can be used together with $S_{\{l_{new}\}}$ to fine-tune $\mathcal{M}_{LA}$ and mitigate CF. Unfortunately, this can also reinforce the errors of $\mathcal{M}_{LA}$, as discussed in Hinton et al. (2015).

Preventing Catastrophic Forgetting. CF is typically caused by the model’s weights, which are pushed towards fitting the data of the latest fine-tuning stage. If the model is not trained using examples in languages $L_A$, it risks forgetting how to treat them. To overcome CF, we propose a method based on Knowledge Distillation (KD). We define a Teacher-Student framework where $\mathcal{M}_{LA}$ acts as the teacher, while the student is a clone of $\mathcal{M}_{LA}$ which is fine-tuned using the multi-loss function $\mathcal{L}_{CL} = \mathcal{L}_T + \mathcal{L}_{KD}$. The term $\mathcal{L}_T$ is the task-specific loss, computed on the annotated examples from $S_{\{l_{new}\}}$. $\mathcal{L}_{KD}$ is a distillation loss computed on $U_{LA}$, a set of unlabeled examples written in the previous languages $L_A$ and here processed by the
teacher model. $L_T$ thus pushes the model to learn how to solve $T$ in the new language $l_{new}$. $L_{KD}$ helps the model maintaining a consistent performance on the languages $L_A$, by forcing the student to mimic the teacher predictions on data resembling the data distribution observed in $L_A$. In order to define $L_{KD}$ consistently with Hinton et al. (2015), let us define $d_i(x)$ as the output logits of the model’s last layer when applied to an example $x$. The logits are converted into a class-probability distribution using the temperature-softmax:

$$
y_i(x) = \frac{\exp(d_i(x)/T)}{\sum_j \exp(d_j(x)/T)}$$

where $T$ is a temperature hyper-parameter, which controls the smoothness of the distribution. $L_{KD}$ is thus computed as the cross-entropy between the output probability distributions provided by the student and teacher, namely $y_i^s$ and $y_i^t$, i.e.:

$$L_{KD}(x) = - \sum_i y_i^s(x) \log y_i^t(x)$$

Using $L_{KD}$ instead of the self-training procedure preserves the uncertainty of the teacher’s model and prevents the student from amplifying the teacher’s errors, as demonstrated in Hinton et al. (2015).

4 Experimental Evaluation

This section presents the results of the proposed CL strategy over three semantic processing tasks, involving text classification and sequence tagging. In particular, we report the Mean Absolute Error (MAE) over the Multilingual Amazon Review Corpus (MARC) (Keung et al., 2020), i.e., a 5 category Sentiment Analysis task in 6 languages. We report the Accuracy over a sentence-pair classification task, i.e., Paraphrase Identification on the PAWS-X dataset (Yang et al., 2019) in 6 languages. Finally, we report the F1 for the Named Entity Recognition (NER) in 4 languages by merging the CoNLL 2002 (Tjong Kim Sang, 2002) and 2003 (Tjong Kim Sang and De Meulder, 2003) datasets. Additional details about the datasets are in Appendix.

Experimental Setup. We foresee a setting where a BERT-based model is incrementally trained using annotated datasets in multiple languages. At each step, the model is fine-tuned using a dataset in one specific language, while the annotated material used up to that point is discarded.

We reasonably assume that a set of unlabeled data is available for the languages already observed. In order to simulate this scenario, we designed a data splitting procedure such that each annotated example is observed only in one step. Let us assume we observe languages in the order $l_1 \rightarrow \ldots \rightarrow l_n$. For each language $l_i$, its training set $D(l_i)$ is divided into $n-i+1$ equal slices, i.e., $(D(l_i), \ldots , D(l_i^{(n-i+1)})$. Depending on the learning strategy, each slice will be either annotated (indicated with a $S$ symbol) or not annotated (indicated with a $U$ symbol). At the last step, we will have observed all the data, either annotated or not.

Learning Strategies. We compare four CL strategies. We denote with CL-Baseline the strategy where at step $k$ the model $M_k$ is obtained by updating $M_{k-1}$ by using only the $S_k = S_{(l_i)}^{(1)}$ annotated dataset, only with the task loss $L_T$. The second strategy is denoted with Self-Training: at step $k$, $M_{k-1}$ is used to annotate the dataset $\tilde{S}_k = \bigcup_{j=1}^{k-1} U_{(l_i)}^{(j)}$. $M_k$ is then fine-tuned by using $S_k = S_{(l_1)}^{(1)} \cup \tilde{S}_k$ with the task loss $L_T$. We denote with CL-KD the strategy we propose, where at step $k$, $M_{k-1}$ is used as the teacher in our proposed KD schema. $M_{k-1}$ is used to derive the target output distribution of the dataset $U_k = \bigcup_{j=1}^{k-1} U_{(l_i)}^{(j)}$. $M_k$ is then trained by adopting $S_k = S_{(l_1)}^{(1)}$, with the task loss $L_T$ and $U_k$ with the loss $L_{KD}$. We compared with a further competitive method, namely Elastic Weight Consolidation, here denoted with EWC (Kirkpatrick et al., 2017). This popular CL procedure applies a regularization technique that penalizes large variations on those model’s weights that are the most important for the tasks learned so far.

As a sort of upper-bound, we report the results by adopting a non-Continual Learning strategy, i.e., Multi-Last, where the model is trained from scratch using an annotated dataset in all languages we want to support at step $k$. More formally, at step $k$ the data is $S_k = \bigcup_{j=1}^{k} S_{(l_i)}^{(k-j+1)}$, i.e., the annotated data is about $k$ times larger than the one used in the CL settings.

We also investigated an approach inspired by Gururangan et al. (2020): we augmented CL-KD with Masked Language Modeling and Next Sentence Prediction objectives to continue the pre-training. Preliminary experiments provided negligible improvements, not reported here due to lack of space.
At each step $k$, we report the average score for the languages that will be observed in steps $(k + 1, \ldots, n)$.

At each step $k$, we report the average score with respect to the languages observed in steps $(1, \ldots, k)$.

Figure 3: Average performance measures on MARC, PAWS-X and CoNLL for the language observed at step $k$.

At each step $k$, we report the average score with respect to the languages observed in steps $(1, \ldots, k)$.

Figure 4: Average performance measures for the MARC, PAWS-X and CoNLL for the languages observed in the past steps. At each step $k$, we report the average score with respect to the languages observed in steps $(1, \ldots, k-1)$.
Model Training. We used the bert-base-multilingual-cased model in the Huggingface Transformers package (Wolf et al., 2019). We trained the models for 10 epochs with Early Stopping (patience = 3) and batch size 32. After initial experiments, we set the temperature $T$ to 1. We repeated our experiments for 6/6/24 sequences of language permutations for MARC/PAWS-X/CoNLL, and we report the average performances.

Experimental Results and Discussion. We first run zero-shot experiments by fine-tuning a model on a subset of languages and testing it on the unobserved ones (see Figure 1). By comparing the results with the ones in Figure 2, we can observe a large gap between the results achieved on the languages still to be observed vs. the training ones. For instance, at step 1 the average gap is more than 30 MAE on MARC, about 0.8% Accuracy on PAWS-X and about 22 F1 on CoNLL. This confirms the need to fine-tune the model on each language of interest.

Figures 2a, 2b and 2c show the results on MARC, PAWS-X and CoNLL, respectively. At each step, we report the average measure computed over all the observed languages, averaged over all the permutations. Given that we are solving the same task in multiple languages, regardless the adopted strategy, the performance can improve at each step due to a cross-lingual transfer learning effect. This beneficial impact is contrasted by the CF, which is also supposed to increase at each step. In our experiments, the effect of transfer learning is generally stronger, with the only exception of CL-Baseline in CoNLL, where CF seems to dominate (the F1 drops from 74.29 at step 1 to 72.67 at step 4). In MARC and PAWS-X, this is alleviated: we argue that CoNLL is more challenging, as it is a word-level tagging on a smaller dataset.

The approach we propose, i.e., CL-KD, is able to constantly outperform its corresponding baseline CL-Baseline. The adoption of knowledge from the previously encountered languages is crucial in mitigating the CF phenomenon. For example, in MARC the MAE in the CL-KD setting is reduced from 60.62 in the first step to 53.85 in the last step. The same applies for PAWS-X where accuracy jumps from 74.57 to 87.73 and for CoNLL with F1 from 74.29 to 79.80. The performances of CL-KD are similar to the Multi-Last even if this clearly has an advantage, using a larger dataset consisting of examples written in all languages.

Figure 3 reports the average performance on the language observed during the last step only, while Figure 4 shows results on the previously acquired languages. Notice that CL-KD achieves comparable results between the previously acquired languages and the last learned one. Conversely, the other CL models perform significantly lower.

Notice that the CL-KD model achieves better results than Self-Training, especially for MARC and CoNLL. This means that classifying the examples with the previous model amplifies the errors of that model. In PAWS-X, the improvements achieved by CL-KD are less evident: we argue this is due to the nature of the dataset, where the training set in each language is derived via automatic machine translation. In any case, CL-KD is still performing better than Self-Training and CL-Baseline: despite automatic translation can be a viable solution, its performances will likely be sub-optimal. Notice that EWC is considered one of the most effective approaches for CL, but interestingly in our setting its results are not satisfactory. We investigated if the order of the languages provides significant differences. We did not notice major variations, also when the involved languages are very different.

Finally, we trained a full-multilingual model with all the data for all the languages. The CL-KD performances are not far from this model, as the difference is only 4.47, 1.56 and 2.44 for MARC, PAWS-X and CoNLL, respectively.

5 Conclusions

This paper investigated a Continual Learning strategy, based on Knowledge Distillation, for training Transformer architectures in an incremental number of languages. We demonstrated that with our approach the model maintains its robustness in processing already acquired languages without having access to annotated data for them, while learning new languages. Future work will apply our methodology to other NLP tasks, such as QA.

Acknowledgments

We would like to thank the “Istituto di Analisi dei Sistemi ed Informatica - Antonio Ruberti” (IASI) for supporting the experimentations through access to dedicated computing resources.

---

3For example, in PAWS-X when ja and zh are the first two languages, the Accuracy at the last step is 87.53. When ja is the third and zh is the fifth, the Accuracy is 87.76. Similar outcomes can be observed for the MARC dataset.
References


A Appendix

A.1 Datasets

Sentence Classification. We used the Multilingual Amazon Reviews Corpus (MARC) (Keung et al., 2020), i.e., a Sentiment Analysis dataset. MARC is a large-scale collection of Amazon reviews in 6 languages (English, German, Spanish, French, Japanese and Chinese). The dataset is made of 200,000/5,000/5,000 reviews for each language, respectively for train, validation and test. We refer to the fine-grained classification (the target category is on 1-5 scale) by using the body of the review.

Sentence-Pairs Classification. We adopted the PAWS-X dataset (Yang et al., 2019) for the Paraphrase Identification task. The dataset is composed of about 24,000 human translated evaluation pairs and about 296,000 machine translated training pairs over 7 languages: English, Spanish, French, German, Japanese, Chinese, Korean. We actually didn’t used the Korean languages, as in preliminary experiment we were not able to reproduce the results of the (Yang et al., 2019) paper. We suspect a problem in the encoding affected our results in this language with the bert multilingual model.

Sequence Tagging. We reported experiments on Named Entity Recognition (NER) using the CoNLL 2002 (Tjong Kim Sang, 2002) and CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) datasets. We merged the two datasets as in Rahimi et al. (2019) to obtain a single dataset over 4 languages, i.e., English, Spanish, German and Dutch. The dataset contains 51,821/11,344/13,556 annotated sentences, respectively for train, validation and test. Each sentence has been annotated with respect to the following entities: Person, Location, Organization and Miscellaneous.

A.2 Additional Results

In this section we report more details on the results of the experiments already discussed in Section 4.

A.2.1 Results on Observed Languages

Tables 1, 2 and 3 complement the results already shown in Figure 2 and summarizes the average performance on the languages observed till each step for MARC, PAWS-X and ConNLL respectively.

<table>
<thead>
<tr>
<th>Step/Model</th>
<th>MULTI-LAST</th>
<th>CL-BASELINE</th>
<th>SELF-TR</th>
<th>EWC</th>
<th>CL-KD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>60.62</td>
<td>60.62</td>
<td>60.62</td>
<td>60.62</td>
<td>60.62</td>
</tr>
<tr>
<td>2</td>
<td>57.77</td>
<td>60.53</td>
<td>62.98</td>
<td>59.55</td>
<td>58.71</td>
</tr>
<tr>
<td>3</td>
<td>56.63</td>
<td>60.59</td>
<td>62.56</td>
<td>60.03</td>
<td>57.39</td>
</tr>
<tr>
<td>4</td>
<td>53.68</td>
<td>58.69</td>
<td>59.83</td>
<td>59.11</td>
<td>55.19</td>
</tr>
<tr>
<td>5</td>
<td>53.70</td>
<td>58.47</td>
<td>59.11</td>
<td>58.33</td>
<td>55.02</td>
</tr>
<tr>
<td>6</td>
<td>51.85</td>
<td>57.30</td>
<td>58.04</td>
<td>57.22</td>
<td>53.85</td>
</tr>
</tbody>
</table>

Table 1: MARC performances for the observed languages (as in Figure 2a), i.e., at each step we report the average of the measure for the languages observed including the last step (step ≤ k). The reported measure is the Mean Absolute Error (lower is better).

<table>
<thead>
<tr>
<th>Step/Model</th>
<th>MULTI-LAST</th>
<th>CL-BASELINE</th>
<th>SELF-TR</th>
<th>EWC</th>
<th>CL-KD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>74.57</td>
<td>74.57</td>
<td>74.57</td>
<td>74.57</td>
<td>74.57</td>
</tr>
<tr>
<td>2</td>
<td>81.78</td>
<td>79.47</td>
<td>78.39</td>
<td>79.62</td>
<td>79.90</td>
</tr>
<tr>
<td>3</td>
<td>84.89</td>
<td>82.34</td>
<td>82.61</td>
<td>82.46</td>
<td>83.42</td>
</tr>
<tr>
<td>4</td>
<td>85.87</td>
<td>84.52</td>
<td>84.24</td>
<td>84.44</td>
<td>85.33</td>
</tr>
<tr>
<td>5</td>
<td>86.81</td>
<td>86.09</td>
<td>85.71</td>
<td>85.98</td>
<td>86.75</td>
</tr>
<tr>
<td>6</td>
<td>87.89</td>
<td>87.17</td>
<td>86.97</td>
<td>87.41</td>
<td>87.73</td>
</tr>
</tbody>
</table>

Table 2: PAWS-X performances for the observed languages (as in Figure 2b), i.e., at each step we report the average of the measure for the languages observed including the last step (step ≤ k). The reported measure is the Accuracy (higher is better).

A.2.2 Results on New Language Only

The following results show how an already fine-tuned model learn to manage a new language. While results in Figure 2 are averaged across all languages (observed up to the k-th step) the following evaluations focus
Table 3: CoNLL 2002/2003 performances for the observed languages (as in Figure 2c), i.e., at each step we report the average of the measure for the languages observed including the last step (step $\leq k$). The reported measure is the F1 (higher is better).

only on the last observed language. Figure 3 and Tables 4, 5 and 6 report the average performance on the last learned language. The average performance tends to improve at each step thanks to the cross-lingual transfer learning effect. All the models perform similarly, exception for the Self-Training model that exhibits generally lower results. This is probably due to the error amplification issue that somehow degrades the cross-lingual transfer.

Table 4: MARC performances for the Current Language (as in Figure 3a). At each step we report the measure for the language observed in that step (step $= k$). The reported measure is the Mean Absolute Error (lower is better).

Table 5: PAWS-X performances for the Current Language (as in Figure 3b). At each step we report the measure for the language observed in that step (step $= k$). The reported measure is the Accuracy (higher is better).

Table 6: CoNLL 2002/2003 performances for the Current Language (as in Figure 3c). At each step we report the measure for the language observed in that step (step $= k$). The reported measure is the F1 (higher is better).

A.2.3 Results on Previously Learned Languages

Figure 4 and Tables 7, 8 and 9 report the average performance for each step on the previously acquired languages. This allows us to better assess the impact of Catastrophic Forgetting. In particular, if we compare these results with the ones reported in Section A.2.2, it is possible to appreciate that model CL-KD achieves comparable results between the previously acquired languages and the last learned one. Conversely, the other CL models, and in particular CL-Baseline, provide significantly lower results on the previously acquired languages w.r.t. to the language learned during the last training step. This is clearly demonstrating the impact of the Catastrophic Forgetting effect.
### A.2.4 Results on Untrained Languages

Figure 1 and Tables 10, 11 and 12 report the average performance for each step on the languages that the model did not train on so far.

<table>
<thead>
<tr>
<th>Step/Model</th>
<th>MULTI-LAST</th>
<th>CL-BASELINE</th>
<th>SELF-TR</th>
<th>EWC</th>
<th>CL-KD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>73.75</td>
<td>73.75</td>
<td>73.75</td>
<td>73.75</td>
<td>73.75</td>
</tr>
<tr>
<td>3</td>
<td>81.46</td>
<td>79.75</td>
<td>78.84</td>
<td>79.71</td>
<td>79.86</td>
</tr>
<tr>
<td>4</td>
<td>84.79</td>
<td>83.04</td>
<td>82.94</td>
<td>82.21</td>
<td>84.07</td>
</tr>
<tr>
<td>5</td>
<td>86.81</td>
<td>85.59</td>
<td>85.19</td>
<td>85.32</td>
<td>86.23</td>
</tr>
<tr>
<td>6</td>
<td>87.68</td>
<td>86.49</td>
<td>86.47</td>
<td>86.37</td>
<td>87.35</td>
</tr>
</tbody>
</table>

Table 11: PAWS-X performances for the Future Languages (zero-shot setting, as in Figure 1b). At each step we report the average of the measure for the languages still not observed (step > k). The reported measure is the Accuracy (higher is better).

This allows us to evaluate the performance of the zero-shot setting. As expected, results are pretty poor,
and the gap between the results on training languages and the zero-shot languages is very large: the gap is more than 30 MAE on MARC, about 8% Accuracy on PAWS-X and about 22 F1 on CoNLL. This confirms the need to fine-tune the model on each language of interest.

<table>
<thead>
<tr>
<th>Step/Model</th>
<th>MULTI-LAST</th>
<th>CL-BASELINE</th>
<th>SELF-TR</th>
<th>EWC</th>
<th>CL-KD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>51.87</td>
<td>51.87</td>
<td>51.87</td>
<td>51.87</td>
<td>51.87</td>
</tr>
<tr>
<td>2</td>
<td>57.88</td>
<td>56.99</td>
<td>57.70</td>
<td>57.15</td>
<td>57.95</td>
</tr>
<tr>
<td>3</td>
<td>61.99</td>
<td>58.57</td>
<td>61.09</td>
<td>59.27</td>
<td>61.86</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 12: CoNLL 2002/2003 performances for the Future Languages (zero-shot setting, as in Figure 1c). At each step we report the average of the measure for the languages still not observed (step > k). The reported measure is the F1 (higher is better).