Abstract

Pretrained multilingual encoders enable zero-shot cross-lingual transfer, but often produce unreliable models that exhibit high performance variance on the target language. We postulate that this high variance results from zero-shot cross-lingual transfer solving an under-specified optimization problem. We show that any linear-interpolated model between the source language monolingual model and source + target bilingual model has equally low source language generalization error, yet the target language generalization error reduces smoothly and linearly as we move from the monolingual to bilingual model, suggesting that the model struggles to identify good solutions for both source and target languages using the source language alone. Additionally, we show that zero-shot solution lies in non-flat region of target language error generalization surface, causing the high variance.

1 Introduction

Pretrained multilingual encoders like Multilingual BERT (mBERT; Devlin et al., 2019) and XLM-RoBERTa (XLM-R; Conneau et al., 2020) facilitate zero-shot cross-lingual transfer (Wu and Dredze, 2019; Hu et al., 2020) — training the model on one language then using it on another language without additional task-specific training data. While the generalization performance on the source language has low variance, on the target language the variance is much higher with zero-shot cross-lingual transfer (Keung et al., 2020; Wu and Dredze, 2020), making it difficult to compare different models in the literature. Similarly, pretrained monolingual encoders also have unstable performance during fine-tuning (Devlin et al., 2019; Phang et al., 2018).

Why are these models so sensitive to the random seed? Many theories have been offered: catastrophic forgetting of the pretrained task (Phang et al., 2018; Lee et al., 2020; Keung et al., 2020), impact of random seed on task-specific layer initialization and data ordering (Dodge et al., 2020), the Adam optimizer without bias correction (Mosbach et al., 2021; Zhang et al., 2021), and a different generalization error with similar training loss (Mosbach et al., 2021). However, none of these factors fully explain the high generalization error variance of zero-shot cross-lingual transfer on target language but low variance on source language.

We offer a new explanation for high variance in target language performance: the zero-shot cross-lingual transfer optimization problem is under-specified. Based on the well-established linear interpolation of 1-dimensional plot and contour plot (Goodfellow et al., 2014; Li et al., 2018), we empirically show that any linear-interpolated model between the monolingual source model and bilingual source and target model has equally low source language generation error. Yet the target language generation error surprisingly reduces smoothly and linearly as we move from a monolingual model to a bilingual model. To the best of our knowledge, no other paper documents this finding.

This result provides a new answer to our mystery: only a small subset of the solution space for the source language solves the target language on par with models with actual target language supervision; the optimization could not find such a solution without target language supervision, hence an under-specified optimization problem. If target language supervision were available, as it was in the counterfactual bilingual model, the optimization finds the smaller subset. By comparing both mBERT and XLM-R, we find that the generalization error surface of XLM-R is flatter than mBERT, contributing to its better performance compared to mBERT. Thus, zero-shot cross-lingual transfer has high variance, as the solution found by zero-shot cross-lingual transfer lies in the non-flat region of the target language generalization error surface.
2 Existing Hypotheses (Related Work)

Prior studies have observed encoder model instability, and have offered various hypotheses to explain this behavior. Catastrophic forgetting—when neural networks trained on one task forget that task after training on a second task (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017)—has been credited as the source of high variance in both monolingual fine-tuning (Phang et al., 2018; Lee et al., 2020) and zero-shot cross-lingual transfer (Keung et al., 2020). Mosbach et al. (2021) wonder why preserving cloze capability is important. In specific task using only source language annotation allows choices of many good solutions in terms of generalization error. However, unbeknownst to the optimizer, these solutions have wildly different generalization errors on the target language. In fact, a small subset has similar low generalization error as models trained on target language. Yet without the guidance of target data, the zero-shot cross-lingual optimization could not find this smaller subset. As we will show in 5, the solution found by zero-shot transfer lies in a non-flat region of target language generalization error, causing its high variance.

3.1 Linear Interpolation

We test this hypothesis via a linear interpolation between two models to explore the neural network parameter space. Consider three sets of neural network parameters: $\theta_{\text{src}}, \theta_{\text{tgt}}, \theta_{\{\text{src,tgt}\}}$ for a model trained on task data for the source language only, target language only and both languages, respectively. This includes both task-specific layers and encoders.\(^1\) Note all three models have the same initialization before fine-tuning, making the bilingual model a counterfactual setup if the corresponding target language supervision was available. We obtain the 1-dimensional (1D) linear interpolation of a monolingual (source) task trained model and bilingual task trained model with

$$\theta(\alpha) = \alpha \theta_{\{\text{src,tgt}\}} + (1 - \alpha) \theta_{\text{src}} \quad (1)$$

or we could swap source and target by

$$\theta(\alpha) = \alpha \theta_{\{\text{src,tgt}\}} + (1 - \alpha) \theta_{\text{tgt}} \quad (2)$$

where $\alpha$ is a scalar mixing coefficient (Goodfellow et al., 2014). Additionally, we can compute a 2-dimensional linear interpolation as

$$\theta(\alpha_1, \alpha_2) = \theta_{\{\text{src,tgt}\}} + \alpha_1 \delta_{\text{src}} + \alpha_2 \delta_{\text{tgt}} \quad (3)$$

where $\delta_{\text{src}} = \theta_{\text{src}} - \theta_{\{\text{src,tgt}\}}, \delta_{\text{tgt}} = \theta_{\text{tgt}} - \theta_{\{\text{src,tgt}\}}, \alpha_1$ and $\alpha_2$ are scalar mixing coefficients (Li et al., 2018).\(^2\) Finally, we can evaluate any interpolated models on the development set of source and target languages, testing the generalization error on the same language and across languages.

---

\(^1\)We experiment with interpolating the encoder parameters only and observe similar findings. On the other hand, interpolating the task-specific layer only has a negligible effect.

\(^2\)Li et al. (2018) use two random directions and they normalize it to compensate scaling issue. In this setup, we find $\delta_{\text{src}}$ and $\delta_{\text{tgt}}$ have near identical norms, so we do not apply additional normalization. As these two directions are not random, we find that it spans around 55\(^\circ\). We plot the norm ratio and angle of these two vectors in App. B.
The performance of the interpolated model illuminates the behavior of the model’s parameters. Take Eq. (1) as an example: if the linear interpolated model performs consistently high for our task on the source language, it suggests that both models lie within the same local minimum of source language generalization error surface. Additionally, if the linear interpolated model performs vastly differently on the target language, it would support our hypothesis. On the other hand, if the linear interpolated model performance drops on the source language, it suggests that both models lie in different local minimum of source language generalization error surface, suggesting the zero-shot optimization searching the wrong region.

4 Experiments

We consider four tasks: natural language inference (XNLI; Conneau et al., 2018), named entity recognition (NER; Pan et al., 2017), POS tagging and dependency parsing (Zeman et al., 2020). We evaluate XNLI and POS tagging with accuracy (ACC), NER with span-level F1, and parsing with labeled attachment score (LAS). We consider two encoders: base mBERT and large XLM-R. For the task-specific layer, we use a linear classifier for XNLI, NER, and POS tagging, and Dozat and Manning (2017) for dependency parsing.

To avoid English-centric experiments, we consider two source languages: English and Arabic. We choose 8 topologically diverse target languages: Arabic, German, Spanish, French, Hindi, Russian, Vietnamese, and Chinese. We train the source language only and target language only monolingual model as well as a source-target bilingual model.

We compute the linear interpolated models as described in §3.1 and test it on both the source and target language development set. We loop over \{-0.5, -0.4, \cdots, 1.5\} for \(\alpha\), \(\alpha_1\) and \(\alpha_2\).  

---

3Arabic is only used when English is the source language.

4We additionally select 0.025, 0.05, 0.075, 0.125, 0.15, 0.175, 0.825, 0.85, 0.875, 0.925, 0.95, and 0.975 for \(\alpha\) due to preliminary experiment.
port the mean and variance of three runs by using different random seeds. We normalized both mean and variance of each interpolated model by the bilingual model performance, allowing us to aggregate across tasks and language pairs. Details of fine-tuning can be found in App. A.

5 Results

In Fig. 1, we observe that interpolations between the source monolingual and bilingual model have consistently similar source language performance. In contrast, surprisingly, the target language performance smoothly and linearly improves as the interpolated model moves from the zero-shot model to bilingual model. The only exception is mBERT, where the performance drops slightly around 0.1 and 0.9 locally. In contrast, XLM-R has a flatter slope and smoother interpolated models.

Fig. 2 further demonstrates this finding with a 2D linear interpolation. The generalization error surface of the target language of XLM-R is much flatter compared to mBERT, perhaps the fundamental reason why XLM-R performs better than mBERT in zero-shot transfer, similar to findings in CV models (Li et al., 2018). As we discuss in §3, these two findings support our hypothesis that zero-shot cross-lingual transfer is an under-specified optimization problem. As Fig. 2 shows, the solution found by zero-shot transfer lies in a non-flat region of target language generalization error surface, causing the high variance of zero-shot transfer on the target language. In contrast, the same solution lies in a flat region of source language generalization error surface, causing the low variance on the source language.

6 Discussion

We have presented evidence that zero-shot cross-lingual transfer is an under-specified optimization problem, and the cause of high variance on target language but not the source language tasks during cross-lingual transfer. This finding holds across 4 tasks, 2 source languages and 8 target languages. Training bigger encoders addresses this issue indirectly by producing encoders with flatter cross-lingual generalization error surfaces. However, a more robust solution may be found by introducing constraints into the optimization problem. Few-shot cross-lingual transfer (Zhao et al., 2021) or silver target data (Yarmohammadi et al., 2021) can provide useful constraints. Unsupervised model selection (Chen and Ritter, 2020) and optimization regularization (Aghajanyan et al., 2021) add constraints without annotation. As none of the existing techniques fully constrain the optimization, future work should study the combination of existing techniques and develop new techniques on top of it.
References


833–844, Hong Kong, China. Association for Computational Linguistics.


A Fine-tuning Experiments Detail

We follow the implementation and hyperparameter of Wu and Dredze (2020). We optimize with Adam (Kingma and Ba, 2014). The learning rate is 2e-5. The learning rate scheduler has 10% steps linear warmup then linear decay till 0. We train for 5 epochs and the batch size is 32. For token level tasks, the task-specific layer takes the representation of the first subword, following previous work (Devlin et al., 2019; Wu and Dredze, 2019). Model selection is done on the corresponding dev set of the training set. We fine-tune each model using a single Quadro RTX 6000 and it takes less than one hour except for XNLI.

During fine-tuning, the maximum sequence length is 128. We use a sliding window of context to include subwords beyond the first 128 for NER and POS tagging. At test time, we use the same maximum sequence length with the exception of parsing, where the first 128 words instead of subwords of a sentence were used. We ignore words with POS tags of SYM and PUNCT during parsing evaluation. For NER, the prediction of BIO was post-processed to make sure a valid span is produced.

All datasets we used are publicly available: NER6, XNLI7, POS tagging and dependency parsing9. For POS tagging and dependency parsing, we use the following treebanks: Arabic-PADT, German-GSD, English-EWT, Spanish-GSD, French-GSD, Hindi-HDTB, Russian-GSD, Vietnamese-VTB, and Chinese-GSD. Data statistic can be found in Tab. 1.

<table>
<thead>
<tr>
<th>Task</th>
<th>NER Examples</th>
<th>POS Examples</th>
<th>Parsing Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>en-train</td>
<td>392703</td>
<td>20000</td>
<td>12543</td>
</tr>
<tr>
<td>en-dev</td>
<td>2490</td>
<td>10000</td>
<td>2002</td>
</tr>
<tr>
<td>ar-train</td>
<td>392703</td>
<td>20000</td>
<td>6075</td>
</tr>
<tr>
<td>ar-dev</td>
<td>2490</td>
<td>10000</td>
<td>909</td>
</tr>
<tr>
<td>de-train</td>
<td>392703</td>
<td>20000</td>
<td>13814</td>
</tr>
<tr>
<td>de-dev</td>
<td>2490</td>
<td>10000</td>
<td>799</td>
</tr>
<tr>
<td>es-train</td>
<td>392703</td>
<td>20000</td>
<td>14187</td>
</tr>
<tr>
<td>es-dev</td>
<td>2490</td>
<td>10000</td>
<td>1400</td>
</tr>
<tr>
<td>fr-train</td>
<td>392703</td>
<td>20000</td>
<td>14449</td>
</tr>
<tr>
<td>fr-dev</td>
<td>2490</td>
<td>10000</td>
<td>1476</td>
</tr>
<tr>
<td>hi-train</td>
<td>392703</td>
<td>5000</td>
<td>13304</td>
</tr>
<tr>
<td>hi-dev</td>
<td>2490</td>
<td>1000</td>
<td>1659</td>
</tr>
<tr>
<td>ru-train</td>
<td>392703</td>
<td>20000</td>
<td>3850</td>
</tr>
<tr>
<td>ru-dev</td>
<td>2490</td>
<td>10000</td>
<td>579</td>
</tr>
<tr>
<td>vi-train</td>
<td>392703</td>
<td>20000</td>
<td>1400</td>
</tr>
<tr>
<td>vi-dev</td>
<td>2490</td>
<td>10000</td>
<td>800</td>
</tr>
<tr>
<td>zh-train</td>
<td>392703</td>
<td>20000</td>
<td>3997</td>
</tr>
<tr>
<td>zh-dev</td>
<td>2490</td>
<td>10000</td>
<td>500</td>
</tr>
</tbody>
</table>

Table 1: Number of examples.

C Normalized Variance of Linear Interpolated Models

Fig. 4 plots the normalized variance of linear interpolated models. We observe that the source language has much lower variance compared to target language on the monolingual side of the interpolated models, echoing findings in Wu and Dredze (2020).

D Break Down of Normalized Performance of Linear Interpolated Models by Tasks

Fig. 5 (NER), Fig. 6 (Parsing), Fig. 7 (POS), and Fig. 8 (XNLI) plot the normalized performance of linear interpolated models break down by task. We observe similar findings as Fig. 1.

E Additional 2D Linear Interpolation

Fig. 9 plots additional 2D linear interpolation. We observe similar findings as Fig. 2.
Figure 3: $\|\delta_{\text{src}}\| / \|\delta_{\text{tgt}}\|$ v.s. angle between $\delta_{\text{src}}$ and $\delta_{\text{tgt}}$. Most $\delta_{\text{src}}$ and $\delta_{\text{tgt}}$ have similar norms, and the angle between them is around $55^\circ$.

Figure 4: Normalized variance of linear interpolation between monolingual model and bilingual model. The source language has much lower variance compared to target language on the monolingual side of the interpolated models.
Figure 5: Normalized NER performance of linear interpolated model between monolingual and bilingual model.

Figure 6: Normalized Parsing performance of linear interpolated model between monolingual and bilingual model.
Figure 7: Normalized POS performance of linear interpolated model between monolingual and bilingual model

Figure 8: Normalized XNLI performance of linear interpolated model between monolingual and bilingual model
Figure 9: Additional normalized performance of 2D linear interpolation between bilingual model and monolingual models.