Language Model Prior for Low-Resource Neural Machine Translation

Christos Baziotis, Barry Haddow and Alexandra Birch

About the paper

A simple approach for incorporating knowledge from monolingual data to NMT.

Uses a LM trained on target side monolingual data, to regularize the output distributions of a TM.

Relevance

- Neural machine translation (NMT) relies heavily on large parallel corpora and needs careful hyperparameter tuning, in order to work in low-resource settings.
- Many low resource languages lack enough parallel data
- Data augmentation techniques like back translation are expensive
  - it requires training separate models and expensive translation of large amounts of monolingual data.
- Exploiting prior information trained on target side data is a principled approach in low resource scenarios.

Other Methods for Incorporating Language Modelling into TM

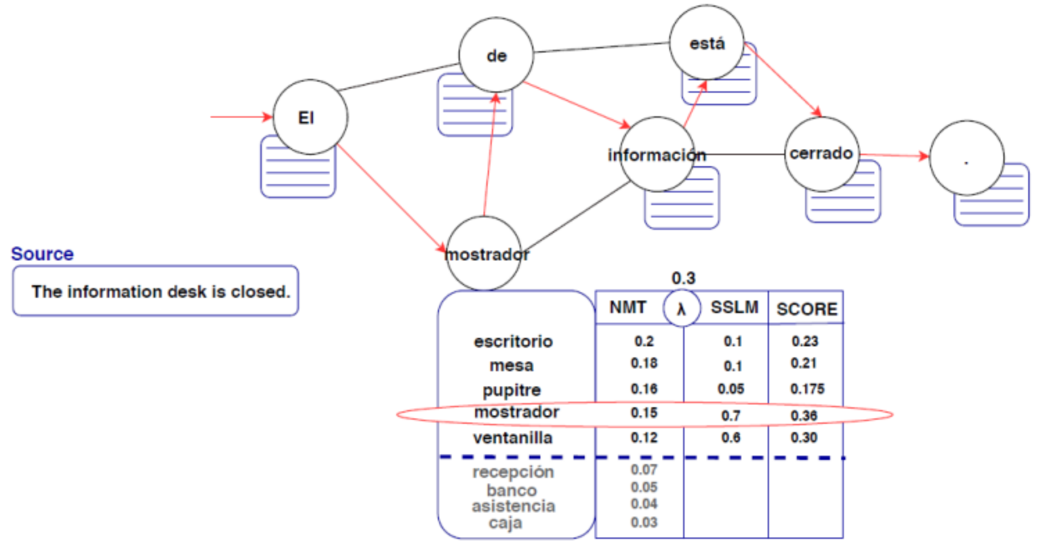
- **Noisy Channel Model – SMT models**
  - It models the "reverse translation probability" $p(x|y)$.
  - It selects words that are both a priori likely with $p(y_i)$ and "explain well" the input with $p(x|y_i)$.
- It has two fundamental limitations.
  - First, during decoding the model has to alternate between generating the output and scoring the input or perform multiple forward passes over x.
  - Second, since the LM is part of the network it has to also be used during inference, which adds a computational constraint on its size.

Previous Methods for incorporating Language Modelling into TM

Fusion

- Incorporate pretrained LMs in NMT, using shallow- and deep-fusion.

- **In shallow-fusion**, the LM re-weights the TM's scores via log-linear interpolation
  - Shallow Fusion combines the probabilities of a translation model and a language model at inference time by introducing a gating mechanism that learns to balance the weight of the additional language model.
  - λ = weight
  - Weight can be adjusted using a grid search on the dev. Data



	NMT	SSLM	SCORE
escritorio	0.2	0.1	0.23
mesa	0.18	0.1	0.21
pupitre	0.16	0.05	0.175
mostrador	0.15	0.7	0.36
ventanilla	0.12	0.6	0.30
recepción	0.07		
banco	0.05		
asistencia	0.04		
caja	0.03		

Figure 1: Sketch of the shallow fusion of an SSLM and an NMT inside the beam search algorithm. In this example, the process re-scores the $N = 5$ best candidates from the NMT model using the scores from the SSLM. Directed edges in the graph mark the path found by the beam search that maximizes the translation probability, whereas undirected edges mark possible steps considered by the beam search algorithm.

Previous Methods for incorporating Language Modelling into TM

- **In deep fusion**, they alter the model architecture to include the hidden states of a RNN-LM as additional features for predicting the next word in the decoder, which are weighted with a controller mechanism (i.e., gating).
- In **deep and shallow fusion** approaches, the TM and LM are first trained independently and are combined later.
- **POSTNORM method** train the TM from scratch with the LM, using its logits as features, instead of its LM hidden states. The LM is used also during training, instead of used just in inference, and interpolating with λ=1.

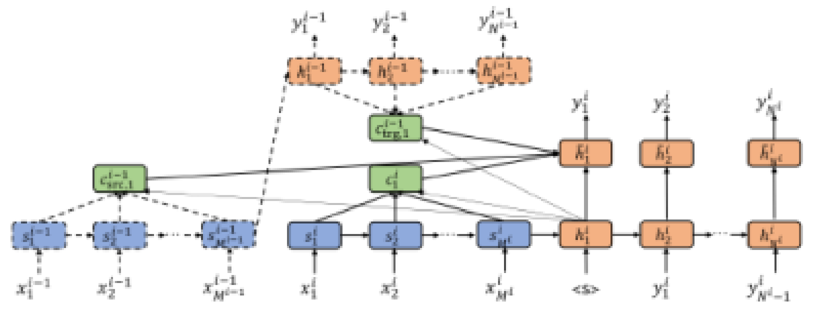
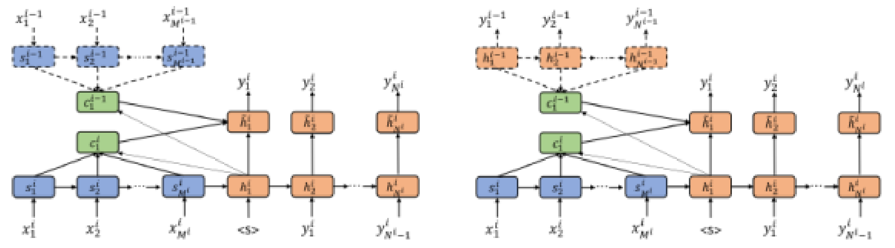


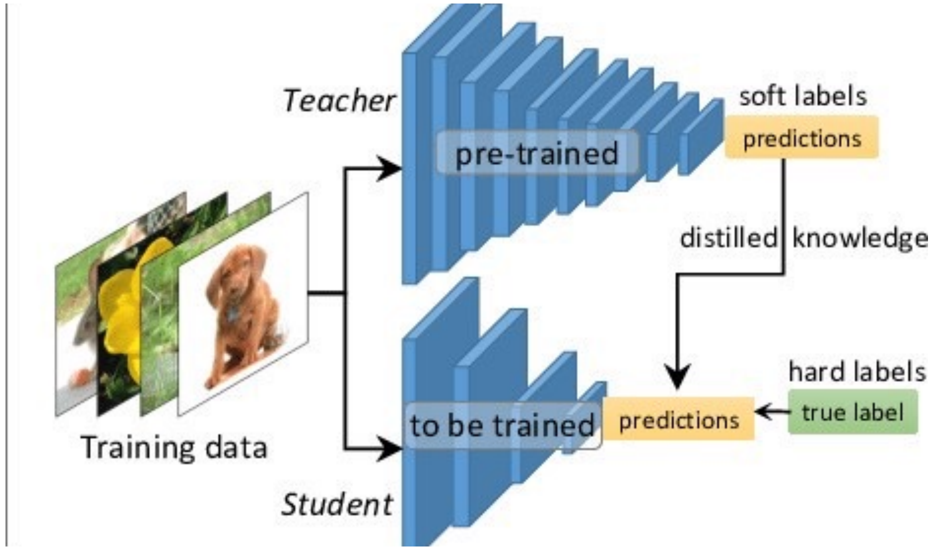
Fig. 2: Shared mix model.



(c) Shared source model. (d) Shared target model.

Contribution

- Incorporate a LM as prior in a neural translation model (TM).
    - Specifically, we add a regularization term, which pushes the output distributions of the TM to be probable under the LM prior, while avoiding wrong predictions when the TM “disagrees” with the LM.
- This objective relates to knowledge distillation, where the LM can be viewed as teaching the TM about the target language.
- The approach does not compromise decoding speed, because the LM is used only at training time, unlike previous work that requires it during inference.
- Show an analysis on the effects that different methods have on the distributions of the TM.
- Results on two low-resource machine translation datasets show clear improvements even with limited monolingual data



Approach

- The LM "teaches" the TM about the target language similar to knowledge distillation.
- It works by simply changing the training objective and does not require any changes to the model architecture.
- Importantly, the LM is separated from the TM, which means that it is needed only during training, therefore we can decode faster than fusion or neural noisy channel.

Model

- Use the LM as a prior over TM's decoder, by employing posterior regularization (PR) (Ganchev et al., 2010).
- Incorporates prior information, by imposing soft constraints on a model's posterior distributions
- The first term is the standard translation objective LMT and the second is the regularization term L_{KL}, over the TM's distributions pTM, that expresses partial information about y.
- L_{KL} is defined as the Kullback-Leibler divergence between the output distributions of the TM and the LM, weighted by λ.
- However, by using the LM-prior we do not change the outputs of the TM. L_{KL} pushes the TM to stay on average close to the prior, but crucially, it enables the TM to deviate from it when needed, for example to copy words from the source.

$$\mathcal{L} = \sum_{t=1}^{N} -\log p_{\mathrm{TM}}(\boldsymbol{y}_t | \boldsymbol{y}_{<t}, \boldsymbol{x}) + \lambda\, D_{\mathrm{KL}}(p_{\mathrm{TM}}(\boldsymbol{y}_t | \boldsymbol{y}_{<t}, \boldsymbol{x}) \parallel p_{\mathrm{LM}}(\boldsymbol{y}_t | \boldsymbol{y}_{<t})) \quad (1)$$

Relation to Knowledge Distillation

In knowledge distillation (KD) the soft output probabilities of a big teacher model are used to train a small compact student model, by minimizing their D_{KL}

Standard KD teacher is trained on the same task as the student

However, the proposed LM-prior is trained on a different task that requires only monolingual data, unlike TM teachers that require parallel data.

Relation to Knowledge Distillation

- Use a softmax temperature parameter   1 to control the smoothness of the output distributions
- $p_i = \frac{\exp(s_i/\tau)}{\sum_j \exp(s_j/\tau)}$, where s_i is the un-normalized score of each word i
- Higher values of produce smoother distributions. Intuitively, this controls how much information encoded in the tail of the LM's distributions, we expose to the TM.
- Specifically, a well-trained LM will generate distributions with high probability for a few words, leaving others with probabilities close to zero. By increasing we expose extra information to the TM, because we reveal more low-probability words that the LM found similar to the predicted word.
- We use $\tau > 1$ only for computing the D_{KL} between the distributions of the TM and the LM.

Relation to Label Smoothing

- Label smoothing (LS) also uses soft targets.
- The purpose of LS is to penalize confidence (i.e., low-entropy distributions).
- LS differs from the LM-prior in two ways.
  - First, LS encourages the model to assign equal probability to all incorrect words, which can be interpreted as a form of uninformative prior (Fig. 1). By contrast, the distributions of the LM are informative, because they express the beliefs of the LM at each step.
  - Second, LS changes the target distribution (i.e., first term in Eq. (1)), whereas the LM-prior involves an additional term, hence the two methods are orthogonal.

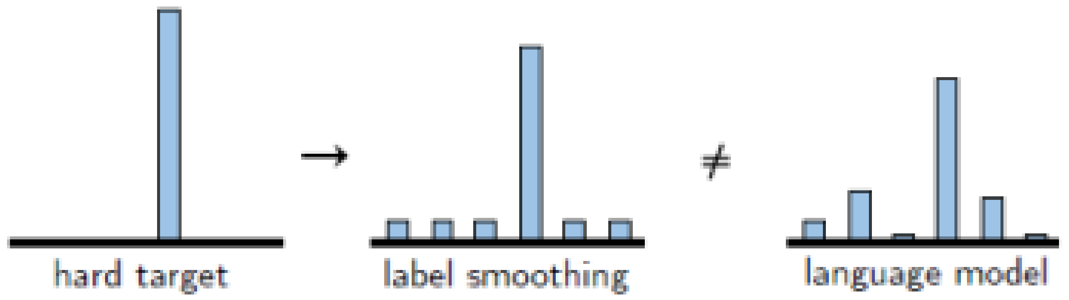
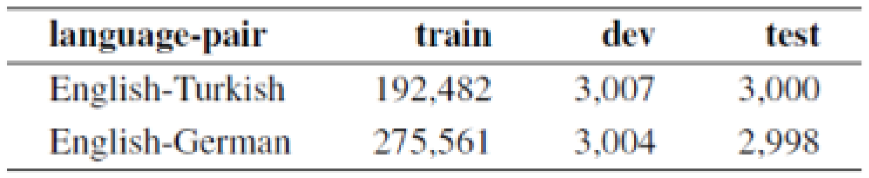


Figure 1: Targets with LS and LM-prior.

Datasets and Preprocessing

- English-German (EN-DE) News Commentary
- English-Turkish (EN-TR) WMT-2018 parallel data from the SETIMES22 corpus.
- Official WMT-2017 and 2018 test sets as the development and test set, respectively.
- Monolingual data
  - English and German News Crawls 2016
  - Turkish concatenate all the available News Crawls data from 2010-2018, which contain 3M sentences.
  - For English and German we subsample 3M sentences to match the Turkish data, as well as 30M to measure the effect of stronger
  - LMs. We remove sentences longer than 50 words.
- Preprocessing
  - Punctuation normalization, truecasing, remove pairs where sentences has more than 60 words or length ratio over 1.5
  - Tokenization with sentence piece with the unigram model
  - A separate model for each language with 16k symbols
  - English is trained on the concatenation of the English side of the training data for each dataset i.e a single vocabulary to be able to reuse LM.

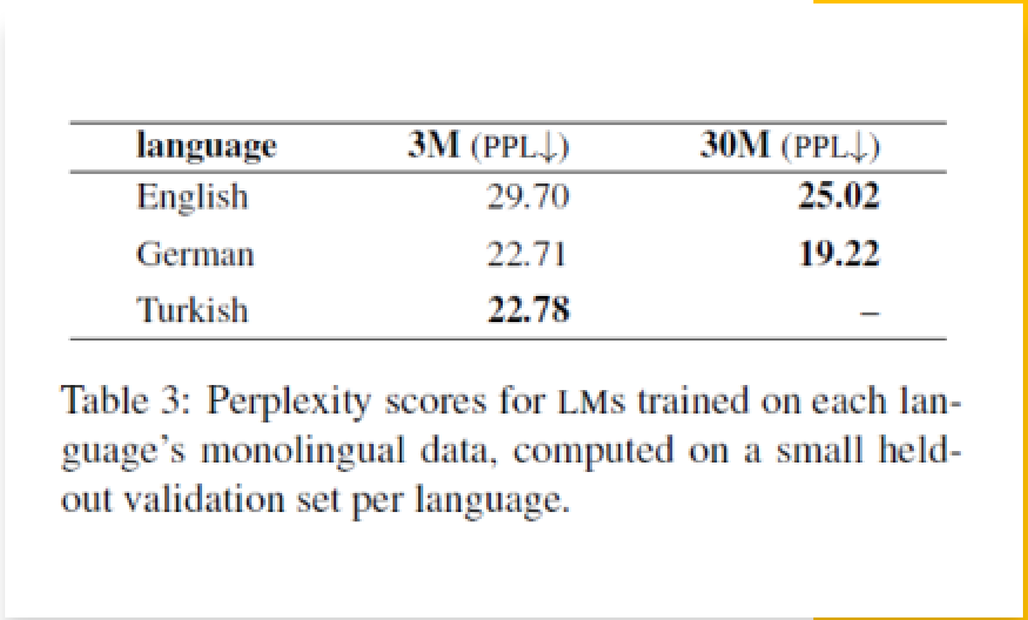


language-pair	train	dev	test
English-Turkish	192,482	3,007	3,000
English-German	275,561	3,004	2,998

Table 1: Dataset statistics after preprocessing.

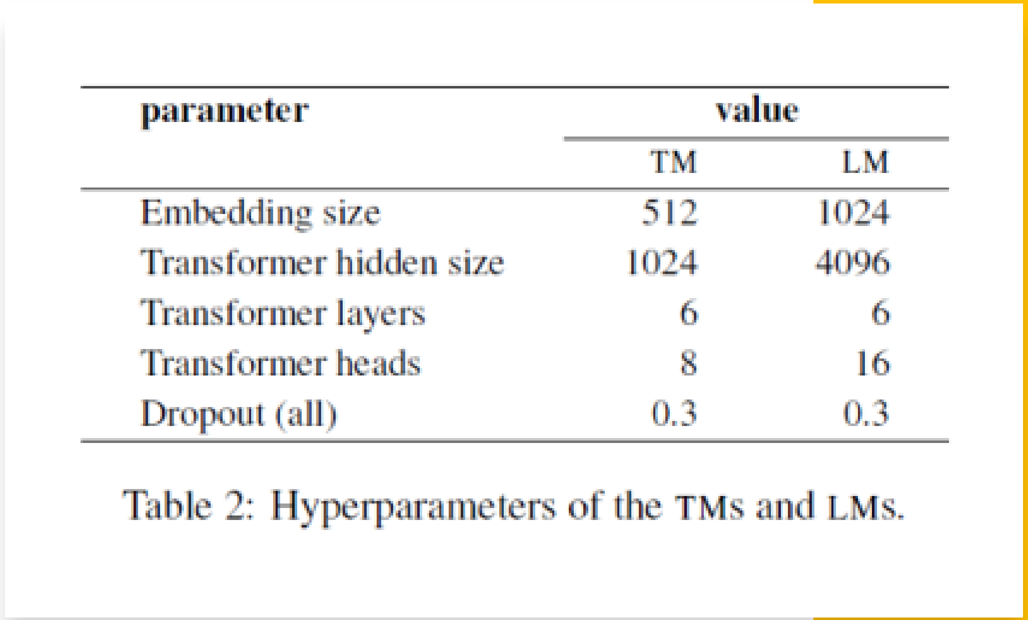
Model configuration

- Transformers pytorch from JoeyNMT
- optimized with Adam
- learning rate of 0.0002
- linear warmup for the first 8K steps
- inverted squared decay
- 5000 tokens per batch.
- We evaluated each model on the dev set every 5000 batches, by decoding using greedy sampling, and stopped training if the
- BLEU score did not increase after 10 iterations.



language	**3M** (PPL↓)	**30M** (PPL↓)
English	29.70	**25.02**
German	22.71	**19.22**
Turkish	**22.78**	–

Table 3: Perplexity scores for LMs trained on each language's monolingual data, computed on a small held-out validation set per language.



parameter	**value**	
	TM	LM
Embedding size	512	1024
Transformer hidden size	1024	4096
Transformer layers	6	6
Transformer heads	8	16
Dropout (all)	0.3	0.3

Table 2: Hyperparameters of the TMs and LMs.

Experiments and Results

- Stronger LMs yield improved results 3M vs 30M
- Adding LM prior yields improvements in low resource scenarios with small models with 10k sentences

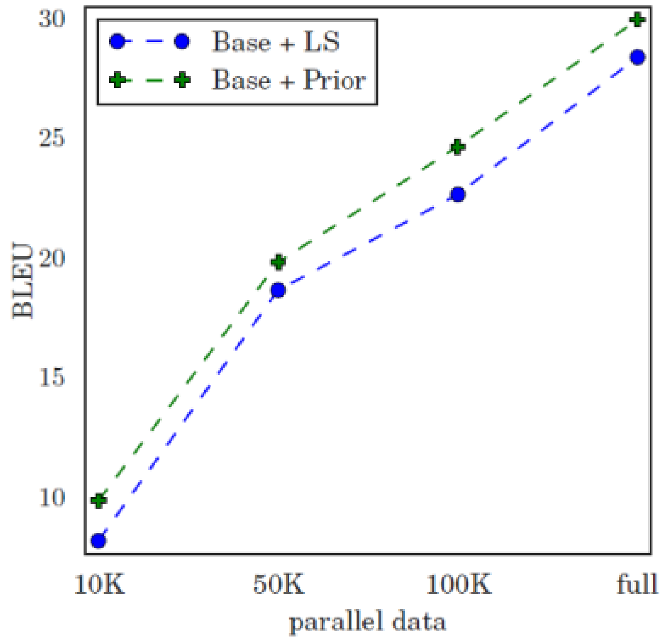
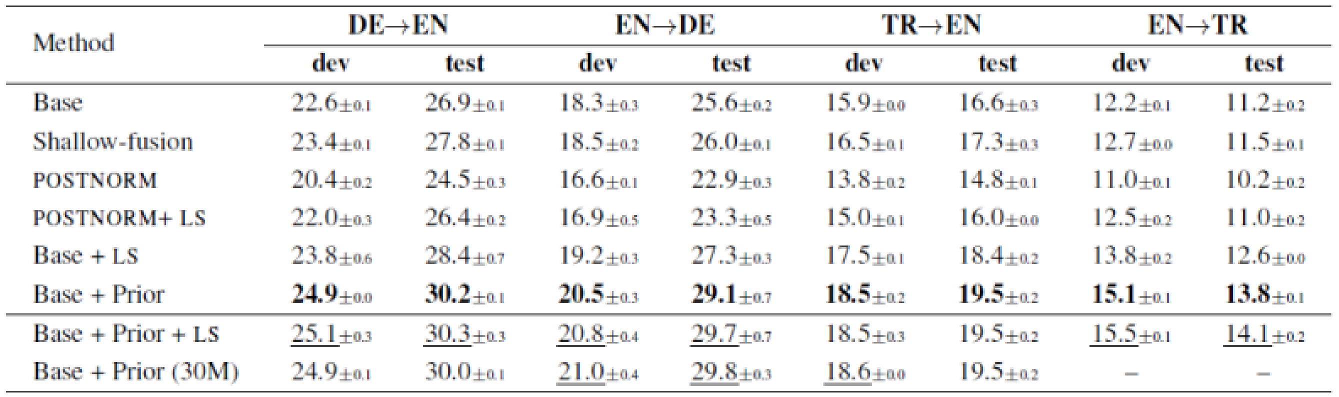


Figure 2: BLEU scores (mean of 3 runs) on the DE→EN test set with different scales of parallel data, using the LM trained on 30M English sentences.



Method	DE→EN		EN→DE		TR→EN		EN→TR	
	dev	test	dev	test	dev	test	dev	test
Base	$22.6_{\pm 0.1}$	$26.9_{\pm 0.1}$	$18.3_{\pm 0.3}$	$25.6_{\pm 0.2}$	$15.9_{\pm 0.0}$	$16.6_{\pm 0.3}$	$12.2_{\pm 0.1}$	$11.2_{\pm 0.2}$
Shallow-fusion	$23.4_{\pm 0.1}$	$27.8_{\pm 0.1}$	$18.5_{\pm 0.2}$	$26.0_{\pm 0.1}$	$16.5_{\pm 0.1}$	$17.3_{\pm 0.3}$	$12.7_{\pm 0.0}$	$11.5_{\pm 0.1}$
POSTNORM	$20.4_{\pm 0.2}$	$24.5_{\pm 0.3}$	$16.6_{\pm 0.1}$	$22.9_{\pm 0.3}$	$13.8_{\pm 0.2}$	$14.8_{\pm 0.1}$	$11.0_{\pm 0.1}$	$10.2_{\pm 0.2}$
POSTNORM+ LS	$22.0_{\pm 0.3}$	$26.4_{\pm 0.2}$	$16.9_{\pm 0.5}$	$23.3_{\pm 0.5}$	$15.0_{\pm 0.1}$	$16.0_{\pm 0.0}$	$12.5_{\pm 0.2}$	$11.0_{\pm 0.2}$
Base + LS	$23.8_{\pm 0.6}$	$28.4_{\pm 0.7}$	$19.2_{\pm 0.3}$	$27.3_{\pm 0.3}$	$17.5_{\pm 0.1}$	$18.4_{\pm 0.2}$	$13.8_{\pm 0.2}$	$12.6_{\pm 0.0}$
Base + Prior	**24.9**$_{\pm 0.0}$	**30.2**$_{\pm 0.1}$	**20.5**$_{\pm 0.3}$	**29.1**$_{\pm 0.7}$	**18.5**$_{\pm 0.2}$	**19.5**$_{\pm 0.2}$	**15.1**$_{\pm 0.1}$	**13.8**$_{\pm 0.1}$
Base + Prior + LS	$\underline{25.1}_{\pm 0.3}$	$\underline{30.3}_{\pm 0.3}$	$\underline{20.8}_{\pm 0.4}$	$\underline{29.7}_{\pm 0.7}$	$18.5_{\pm 0.3}$	$19.5_{\pm 0.2}$	$\underline{15.5}_{\pm 0.1}$	$\underline{14.1}_{\pm 0.2}$
Base + Prior (30M)	$24.9_{\pm 0.1}$	$30.0_{\pm 0.1}$	$\underline{21.0}_{\pm 0.4}$	$\underline{29.8}_{\pm 0.3}$	$\underline{18.6}_{\pm 0.0}$	$19.5_{\pm 0.2}$	–	–

Table 4: BLEU scores of each model. Mean and stdev of 3 runs reported. The top section contains the main results, where all methods use LMs trained on the *same* amount of data (3M). The bottom section compares different configurations of the LM-prior. Underlined scores denote gains over the "Base + Prior (3M)" model.

Analysis

- LS, that simply penalizes confidence, is a very effective form of regularization in low-resource settings. It makes the TM less confident and therefore more robust to over-fitting.
- Lmprior emits more confident distributions than the “Base+LS” model
- Base+POSTNORM model and observe that it generates the most confident predictions.
    - only a small subset of words will have non-zero probability in the final distribution. This means that when there are “disagreements” between the TM and LM this can lead to wrong predictions.

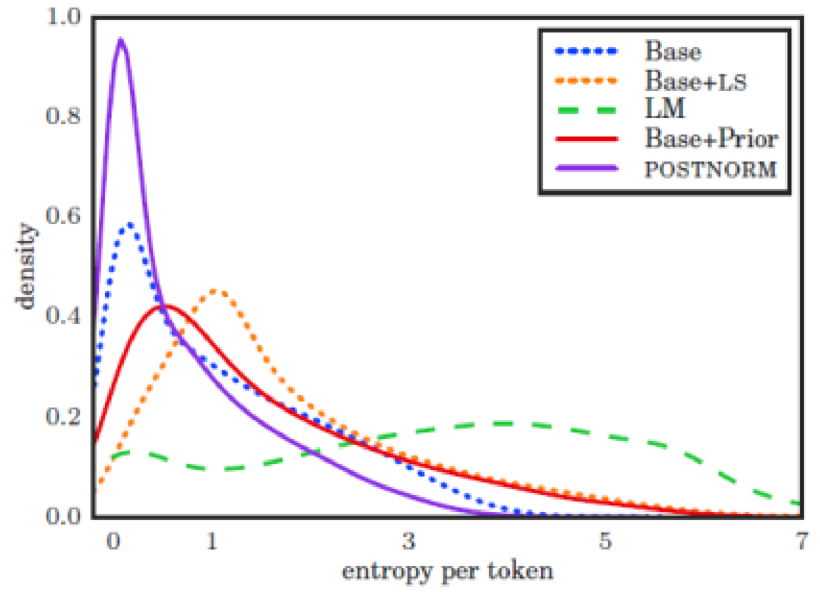
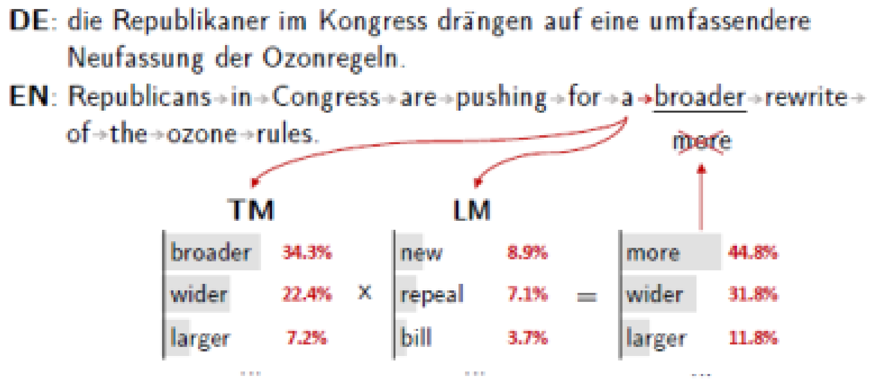


Figure 3: Estimated densities based on each model’s entropy on the DE→EN test set.



DE: die Republikaner im Kongress drängen auf eine umfassendere Neufassung der Ozonregeln.

EN: Republicans in Congress are pushing for a broader rewrite of the ozone rules.

~~more~~

TM			LM				
broader	34.3%		new	8.9%		more	44.8%
wider	22.4%	×	repeal	7.1%	=	wider	31.8%
larger	7.2%		bill	3.7%		larger	11.8%
...			...			...	

Figure 4: Example of failure of probability interpolation between LM and TM, while translating DE→EN.

Sensitivity Analysis

- Using $\tau > 1$ helps the TM to acquire more of the knowledge encoded in the prior,
- Increasing the strength of the regularization up to a point yields consistent improvements.
- We find that the performance is less
- sensitive to the value of τ, compared to λ by setting $\tau > 1$, the model becomes also more robust to λ.

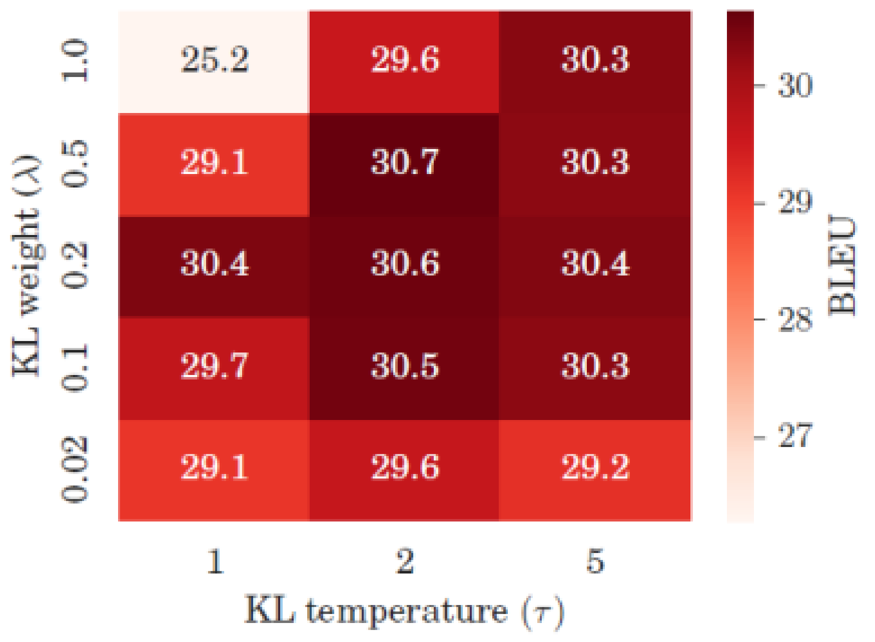


Figure 5: BLEU scores on the DE-EN dev set of models trained with different λ and τ for the $\mathcal{L}_{\text{KL}}$. Mean of 3 runs for each combination reported.

Conclusion

This method is more efficient than alternative approaches that used pretrained LMs, because it is not required during inference.

Also, it avoids the translation errors introduced by LM-fusion, because the TM is able to deviate from the prior when needed. We empirically show that while this method

It works by simply changing the training objective, it achieves better results than alternative LM-fusion techniques.

Also, it yields consistent performance gains even with modest monolingual data (3M sentences) across all translation directions.

This makes it useful for low-resource languages, where not only parallel but also monolingual data are scarce.