The Trade-offs of Domain Adaptation for Neural Language Models

David Grangier
Google, Mountain View, CA
grangier@google.com

Dan Iter[*]
Stanford, Palo Alto, CA
daniter@stanford.edu

Abstract

This work connects language model adaptation with concepts of machine learning theory. We consider a training setup with a large out-of-domain set and a small in-domain set. We derive how the benefit of training a model on either set depends on the size of the sets and the distance between their underlying distributions. We analyze how out-of-domain pre-training before in-domain fine-tuning achieves better generalization than either solution independently. Finally, we present how adaptation techniques based on data selection, such as importance sampling, intelligent data selection and influence functions, can be presented in a common framework which highlights their similarity and also their subtle differences.

1 Introduction

Neural Language Models (LMs) trained on large generic training sets – over a billion sentences (Kaplan et al., 2020; Roziewski and Kozłowski, 2021) – have been shown to be effective at adapting to smaller, specific target domains for language modeling and other downstream tasks (Bommasani et al., 2021). Neural LM adaptation is commonly performed via fine tuning (Devlin et al., 2018; Liu et al., 2019; Raffel et al., 2019; Radford et al., 2019), data selection (van der Wees et al., 2017) or their combination (Wang et al., 2018; Aharoni and Goldberg, 2020; Gururangan et al., 2020). However, the trade-offs between fine-tuning and reweighting of pre-training data is not well understood and a theoretical framework for reasoning about the generalization performance of these methods is needed.

In this paper, we connect language model adaptation with concepts of machine learning theory. Our derivations support past empirical observations: it has been observed that the size of the out-of-domain pre-training set is important for in domain generalization (Raffel et al., 2019; Devlin et al., 2018) or that domain adaptation is more effective on domains which are well represented in the the pre-training data (Radford et al., 2019). Our study consider a training setup with a large out-of-domain set and a small in-domain set. As a first contribution, we derive how the benefit of training a model on either set depends on the size of the sets and the distance between their underlying distribution. We also expose how fine-tuning can be viewed as a regularization method that can achieve a better trade-off than training only on either set.

The research on data selection for LM adaption originates mainly from intelligent selection (Moore and Lewis, 2010; Axelrod et al., 2011). This method examines the out-of-domain training data to emphasize a subset deemed more likely by an in-domain model than by an out-of-domain model. Although intuitive, the connection of this method with statistical estimation is unclear, which makes studying its impact on generalization error difficult. Another family of selection methods stems from influence functions (Koh and Liang, 2017; Wang et al., 2021) which estimate whether the model updates from out-of-domain training examples are aligned with the in-domain updates. This approach is more principled and its impact on the generalization error is easier to study. In this work, as a second contribution, we show how intelligent selection and influence function methods are linked in the case of neural LMs. In particular, we show that they both can be derived from importance sampling (Owen, 2013), a classical, well-studied statistical estimation technique.

The rest of our paper is organized as follows. We first presents the theoretical trade-offs between in-domain and out-of-domain training. We highlight the importance of the relative sizes of in-domain and out-of-domain training sets along with

[*]Work performed while interning at Google.

the distance between their underlying distributions. We also present how fine-tuning with a limited number of updates can be seen as a training method regularized with respect to the out-of-domain prior. Finally, we present data selection methods under a unifying framework.

2 Neural Language Modeling

Language modeling refers to the generative modeling of natural language (Manning and Schutze, 1999). Commonly, natural language is represented as a sequence of symbols, tokens, from a finite vocabulary. For instance, language can be represented as a sequence of characters, a sequence of words or alternative units. A neural language model (LM) decomposes the estimates the log probability of a text $y = (y_1, \dots y_n)$, as

$$\log P(y; \theta) = \sum_{i=1}^{n} \log P(y_i | y_1^{i-1}; \theta)$$

where P_θ maps a parameter vector θ along with a sequence of past tokens y_1^{i-1} onto a probability distribution over the vocabulary. Different types of neural architectures have been used for neural language modeling. Most architectures used for LMs re-use intermediate computations from the previous steps for the next steps when estimating probabilities for successive tokens in the same sequence. Popular architectures include recurrent neural networks (Mikolov et al., 2010; Sundermeyer et al., 2012), convolutional networks (Dauphin et al., 2017) and transformer networks (Vaswani et al., 2017; Radford et al., 2019).

The parameter vector $\theta \in \Theta$ of a neural LM is identified by maximizing the log likelihood over a training set D sampled from the true distribution $\mathcal{D}$ using variants of stochastic gradient descent. The log likelihood of a held-out set, sampled from the same distribution, can evaluate model quality. One often reports perplexity, the exponentiated negative average log likelihood per token.

Conditional LMs model the distribution of a text y given a conditioning input x.

$$\log P(y|x; \theta) = \sum_{i=1}^{n} \log P(y_i | y_1^{i-1}, x; \theta)$$

This type of model is used for translation where (x, y) pairs are sentences in the source and target language (Koehn, 2009; Bahdanau et al., 2015) or summarization where (x, y) pairs are corresponding articles and summaries (See et al., 2017).

For both conditional and regular LMs, the size of the training data is important to achieve a low held-out perplexity. This is an obstacle for domains with limited available training data. This issue has led to various model adaptation approaches. These methods leverage large amounts of generic training data D along with a small amount of target domain training data T from the domain of interest. *Fine tuning* is a popular domain adaptation method which trains a neural language model in two phases, first maximizing the likelihood of the generic set D (pre-training) before optimizing the likelihood of the target domain set T (fine-tuning). As an alternative to fine-tuning, some methods consider leveraging the small target-domain training set to identify and emphasize similar data in the larger generic training set. These *emphasis* methods can be used individually or in conjunction with fine-tuning. Emphasis methods include importance sampling, contrastive data selection and influence functions. This paper shows that these methods – although proposed in different context – can be presented in a unified way which allows light to be cast on their subtle differences.

3 Training Strategies

This section first examines in-domain training, i.e. when the training and test data are sampled from the same distribution. It then studies out-of-domain training, i.e. when the training and test data distribution differs. Finally, it examines out-of-domain pre-training followed by in-domain fine tuning. For the three cases, we decompose the loss relying on classical concepts from learning theory and study the trade-offs involved in each setup.

3.1 In-Domain Training

Given a training set D sampled from a distribution $\mathcal{D}$, learning an LM typically aims at minimizing the negative log-likelihood of D, also referred to as the *cross-entropy* loss i.e.

$$\mathcal{L}(\theta; D) = -\frac{1}{|D|} \sum_{y \in D} \log P(y|\theta) = \mathop{\mathbb{E}}_{y \sim D} [-\log P(y|\theta)].$$

This *empirical risk* is the average over the finite set D, which acts as a proxy for the expectation over the true, unavailable distribution $P(y|\mathcal{D})$,

$$\begin{aligned}\mathcal{L}(\theta; \mathcal{D}) &= -\sum_{y \in \Omega} \log P(y|\theta) P(y|\mathcal{D}) \\ &= \mathop{\mathbb{E}}_{y \sim \mathcal{D}} [-\log P(y|\theta)],\end{aligned}$$

where the distribution's support Ω is the set of all finite sequences. The true expected loss is bounded by the entropy of the distribution $P(\cdot|\mathcal{D})$, i.e.

$$\mathcal{L}(\theta; \mathcal{D}) \geq \mathcal{L}_H(\mathcal{D}) = H(P(\cdot|\mathcal{D}))$$

since $H(P(\cdot|\mathcal{D})) = \min_q \mathbb{E}_{y \sim \mathcal{D}}[-\log q(y)]$. The gap between the best likelihood from a neural network with the chosen parameterization and the entropy is called the approximation error

$$\mathcal{L}_{\text{app}}(\mathcal{D}, \Theta) = \min_{\theta \in \Theta} \mathcal{L}(\theta; \mathcal{D}) - H(P(\cdot|\mathcal{D})).$$

This gap accounts for the fact that $P(\cdot|\mathcal{D})$ generally cannot be represented by a parameterized function from the chosen family spanned by Θ. In addition to the approximation error, one should consider the estimation error to account that one relies on the empirical risk from the finite set D,

$$\mathcal{L}_{\text{est}}(\mathcal{D}, \Theta, D) = \mathcal{L}(\theta_D; \mathcal{D}) - \min_{\theta} \mathcal{L}(\theta; \mathcal{D})$$

with $\theta_D = \arg\min_{\theta \in \Theta} \mathcal{L}(\theta; D)$. Therefore, the loss of θ_D over $\mathcal{D}$ decomposes as (Bottou and Bousquet, 2007)

$$\boxed{\begin{aligned}&\mathcal{L}(\theta_D; \mathcal{D}) = \\ &\quad \mathcal{L}_H(\mathcal{D}) + \mathcal{L}_{\text{app}}(\mathcal{D}, \Theta) + \mathcal{L}_{\text{est}}(\mathcal{D}, \Theta, D)\end{aligned}} \quad (1)$$

where the three terms accounts for the intrinsic uncertainty of $\mathcal{D}$, the chosen neural architecture and the finite training set D respectively.

The approximation error $\mathcal{L}_{\text{app}}(\mathcal{D}, \Theta)$ depends on the selected model family Θ. It can be reduced by selecting a more expressive family, i.e. a neural architecture with more capacity, a larger Θ, e.g. architectures with more, wider layers. The estimation error $\mathcal{L}_{\text{est}}(\mathcal{D}, \Theta, D)$ depends both on the selected model family Θ and the size of the training data D. Increasing model capacity will result in a higher estimation error for the same training set size, but training over a larger training set will decrease estimation error. Therefore, for a given training set size, capacity needs to be chosen to identify a good trade-off between the two error types.

Two important properties of neural networks need to be kept in mind when examining this trade-off. The *universal approximation* property (Lecun, 1987; Funahashi, 1989) means that for any approximation error ϵ and any distribution $\mathcal{D}$, there exists a capacity setting $C(\epsilon, \mathcal{D})$ at which a neural network $\theta \in C(\epsilon, \mathcal{D})$ whose error is below ϵ, i.e.

$$\forall \epsilon > 0, \exists\, C \text{ s.t. } \mathcal{L}_{\text{app}}(\mathcal{D}, C) \leq \epsilon.$$

In layman terms, the *universal approximation* property means that for sufficiently large capacity settings, the approximation error can become arbitrary low. The *statistical consistency* property means that for any $\epsilon, \epsilon' > 0$, there exist a training set size $N(\epsilon, \mathcal{D})$ such that sampling a training set of size $N(\epsilon, \epsilon', \mathcal{D})$ from $\mathcal{D}$ will result in an estimation error less than ϵ' with probability $1 - \epsilon$, $\forall \epsilon, \epsilon' > 0, \exists\, N$ s.t ,

$$P(D \sim \mathcal{D}^N : \mathcal{L}_{\text{est}}(\mathcal{D}, \Theta, D) < \epsilon') = 1 - \epsilon$$

In layman terms, the *statistical consistency* property means that for sufficiently large training sets, the probability to get an estimation error below any positive value can be arbitrary close to 1.

Universal approximation and consistency implies that, in the asymptotic case (i.e. as the size of D tends to infinity), the last two terms in Eq. 1 can be arbitrary close to zero with the appropriate model capacity (with high probability). In that case, the likelihood $\mathcal{L}(\theta_D; \mathcal{D})$ amounts to the intrinsic entropy of $\mathcal{D}$ with the appropriate model capacity.

3.2 Out-of-Domain Training

This section considers a setup where one needs a specialized language model in a domain $\mathcal{T}$ and two training sets are available: a small training set T sampled from $\mathcal{T}$ and a large training set D sampled from $\mathcal{D}$, a generic domain different from the specialized domain.

In that context, the simplest options are either to train a model over T or D alone. Training only on the small set T results in the generalization loss

$$\begin{aligned}&\mathcal{L}(\theta_T; \mathcal{T}) \\ &\quad = \mathcal{L}_H(\mathcal{T}) + \mathcal{L}_{\text{app}}(\mathcal{T}, \Theta) + \mathcal{L}_{\text{est}}(\mathcal{T}, \Theta, T)\end{aligned}$$

with $\theta_T = \arg\min_{\theta \in \Theta} \mathcal{L}(\theta; T)$ as in the previous section. Training on the larger set D results in

$$\begin{aligned}&\mathcal{L}(\theta_D; \mathcal{T}) \\ &\quad = \mathcal{L}_H(\mathcal{T}) + \mathcal{L}_{\text{app}}(\mathcal{T}, \Theta) + \mathcal{L}_{\text{est}}(\mathcal{T}, \Theta, D).\end{aligned}$$

Two factors are important to compare these two options: the size of the specialized set T relative to the size of the generic set D and the similarity between $\mathcal{T}$ and $\mathcal{D}$ distributions.

When the $\mathcal{T}$ and $\mathcal{D}$ distributions are identical, D and T are sampled from the same distribution and training a model on the larger training set D is advantageous. For a constant capacity, this option will get a lower estimation error. When varying capacity, one might identify a setting with an even better trade-off in the compound loss of Eq. (1) with the larger training set D.

When the distributions $\mathcal{T}$ and $\mathcal{D}$ differ and the size of D is fixed, the size of T determines which option to prefer. Statistical consistency means that $\mathcal{L}_{\text{est}}(\mathcal{T}, \Theta, T)$ will converge to zero in probability as the size of T grows. This means that when the size of T is greater than $N(\epsilon, \mathcal{L}_{\text{est}}(\mathcal{T}, \Theta, D), \mathcal{D})$, the probability that training on T results in a better generalization loss than training on D is above $1 - \epsilon$.

When the distributions $\mathcal{T}$ and $\mathcal{D}$ differ, the Kullback–Leibler (KL) divergence between the two distributions plays a key role.

Theorem 1 The generalization of the loss of θ_D over T is upper bounded as

$$\boxed{\begin{aligned}&\forall \epsilon > 0,\ \exists N \text{ s.t. } \forall D \sim \mathcal{D}^n,\\ &\quad \mathcal{L}(\theta_D; \mathcal{T}) \leq H(\mathcal{T}) + KL(\mathcal{T}, \mathcal{D}) + \epsilon\end{aligned}} \quad (2)$$

with probability $1 - \epsilon$. This bound justifies the intuition that, if given the choice between two generic domains $\mathcal{D}$ and $\mathcal{D}'$, training over the one with the lowest KL divergence to $\mathcal{T}$ will result in a better asymptotic behaviour. The proof of this bound is presented in Appendix A.

3.3 Fine-Tuning & Multitask Learning

Fine-tuning for domain adaptation trains a model on a small in-domain set initializing optimization from the parameters of a model trained on a large out-of-domain set. Formally, fine-tuning minimizes $\mathcal{L}(\theta; T)$ the loss over T for a few steps, starting the optimization from $\theta_D = \arg\min_{\theta\in\Theta} \mathcal{L}(\theta; D)$. This strategy implicitly targets a trade-off between the empirical losses over T and D. This trade-off is controlled by the number of fine tuning steps n_{ft}. Few steps means that the identified parameters θ_{ft} achieve a low loss over D, while many steps expresses that the parameters achieve a low loss over T. This strategy leverages the regularization effect of early stopping (Caruana et al., 2001), i.e. the solution found by gradient descent is guaranteed to be in an Euclidean ball centered around the initialization whose radius grows with the number of steps (Grangier and Bengio, 2008), i.e.

$$\|\theta_{\text{ft}} - \theta_D\|_2 \leq \lambda\, n_{\text{ft}}\, g_{\max}$$

where λ refers to the (maximum) learning rate and $g_{\max}$ to an upper bound on the update norm. The small distance between θ_{ft} and θ_D guarantees that the loss $\mathcal{L}(\theta_{\text{ft}}; D)$ is close to the optimum $\mathcal{L}(\theta_D; D)$ when $\theta \to \mathcal{L}(\theta; D)$ is a smooth function, e.g. a Lipschitz function.

For the basic fine-tuning setup, several variants have been introduced. Some approaches (Devlin et al., 2018; Liu et al., 2019; Raffel et al., 2019) consider leaving some parameters un-tuned or frozen which is the extreme case of regularization for these weights, penalizing any deviation from initialization. Other approaches consider introducing novel (unregularized) weights for fine tuning, often referred as *adapter* layers (Houlsby et al., 2019; Stickland et al., 2019; Pfeiffer et al., 2020). Other forms of regularization, such as dropout, have also been considered in conjunction with fine tuning (Miceli Barone et al., 2017).

The selection of the regularization strength in fine-tuning is computationally efficient since it successively visits an optimization path from the most regularized model (θ_D trained only on D, Sec. 3.2) to the unregularized θ_T (Sec. 3.1). This is more efficient compared to explicit regularization methods, including multitask learning (Caruana, 1998; Collobert and Weston, 2008; Pilault et al., 2021), i.e. optimizing $\mathcal{L}_{\text{multi}}(\theta; T, D, \alpha) = \mathcal{L}(\theta; T) + \alpha\mathcal{L}(\theta; D)$.

4 Data Selection

Data selection aims to improve out-of-domain training by selecting or giving stronger weights to some data points. The identification of these points aims to emphasize out-of-domain examples which have an impact on the model similar to the impact of the in-domain training examples. We study three independently proposed selection methods, importance sampling, contrastive data selection and influence functions. We show that these methods all train models through weighted log-likelihood training,

$$\mathcal{L}(\theta; D, T, w) = -\frac{1}{|D|}\sum_{y\in D} w(y; \mathcal{T}, \mathcal{D}) \log P(y|\theta)$$

but introduce their weights $w(y; \mathcal{T}, \mathcal{D})$ with different justifications. Despite these differences, we show that these methods result in surprisingly similar selection weights in the specific case of neural language models.

Data selection is particularly suited when the out-of-domain training distribution and the test distribution have a large KL divergence but the out-of-domain training set is large. In that case, the generalization of a model trained on out-of-domain data is poor due to the large KL divergence between $\mathcal{T}$ and $\mathcal{D}$, see Eq. (2). When this KL divergence is large but out-of-domain data is abundant, data selection methods propose to select a subset of the out-of-domain data $D^{\mathcal{T}} \subset D$. Ideally, the training loss over such a subset $\mathcal{L}(\theta, D^{\mathcal{T}})$ would be a better proxy for the generalization loss over $\mathcal{T}$, $\mathcal{L}(\theta, \mathcal{T})$, than the training loss over the full set D, $\mathcal{L}(\theta, D)$.

Selection involves a delicate trade-off though. One one hand, data selection is attractive since it replaces the training set with another set closer to the test domain. On the other hand, this training set is smaller, which increases the impact of estimation errors. Additionally, data selection is imperfect since the target domain distribution $\mathcal{T}$ is only known through a small target training set T.

This section successively presents importance sampling, contrastive data selection and influence functions and connect them into a single framework.

4.1 Importance Sampling

Although intelligent selection also called contrastive data selection is more common (Moore and Lewis, 2010; Wang et al., 2018), we first examine importance sampling since this method will guide our understanding of other selection methods.

Importance sampling is a generic statistical technique (Owen, 2013). In our case, it can be used to estimate the expectation of the cross-entropy loss over $\mathcal{T}$ while having access to samples from $\mathcal{D}$. It relies on the identity

$$\begin{aligned}
\mathcal{L}(\theta; \mathcal{T}) &= \mathop{\mathbb{E}}_{y\sim\mathcal{T}}[-\log P(y|\theta)] \\
&= -\sum_{y\in\Omega} \log P(y|\theta) P(y|\mathcal{T}) \\
&= -\sum_{y\in\Omega} \log P(y|\theta) \frac{P(y|\mathcal{T})}{P(y|\mathcal{D})} P(y|\mathcal{D}) \\
&= \mathop{\mathbb{E}}_{y\sim\mathcal{D}}[-w(y; \mathcal{T}, \mathcal{D}) \log P(y|\theta)]
\end{aligned}$$

where $w(y; \mathcal{T}, \mathcal{D}) = \frac{P(y|\mathcal{T})}{P(y|\mathcal{D})}$, assuming full support on $\mathcal{D}$, i.e. $\forall y \in \Omega$, $P(y|\mathcal{D}) > 0$. In practice, one has not access to $\mathcal{T}$ and $\mathcal{D}$ but to finite samples T and D. With importance sampling, we can consider two alternative estimators of $\mathcal{L}(\theta; \mathcal{T})$, either the empirical risk over T,

$$\mathcal{L}(\theta; T) = -\frac{1}{|T|} \sum_{y\in T} \log P(y|\theta)$$

or the mean of the importance weighted cross entropy over D, i.e.

$$\mathcal{L}_{\text{imp}}(\theta; D, T, \hat{w}) = -\frac{1}{|D|} \sum_{y\in D} \hat{w}(y; \mathcal{T}, \mathcal{D}) \log P(y|\theta)$$

where $\hat{w}$ estimates of the weights w from the training sets D and T. The trade-off between these two estimators depends on the relative size of T and D, the imbalance of the weights w and the quality of their estimate $\hat{w}$.

Importance sampling is interesting when the generalization error $\mathcal{L}(\theta_{\text{imp}(D,T)}; \mathcal{T})$ of the model

$$\theta_{\text{imp}(D,T)} = \arg\min_{\theta} \mathcal{L}_{\text{imp}}(\theta; D, T, \hat{w})$$

is less than the generalization error of θ_T selected by minimizing $\mathcal{L}(\theta; T)$, i.e. classical empirical risk minimization. This error decomposes as,

$$\begin{aligned}
&\mathcal{L}(\theta_{\text{imp}(D,T)}; \mathcal{T}) \\
&= \mathcal{L}_H(\mathcal{T}) + \mathcal{L}_{\text{app}}(\mathcal{T}, \Theta) + \mathcal{L}^{\text{imp}}_{\text{est}}(\mathcal{T}, \Theta, D, T).
\end{aligned}$$

We further decompose the estimation error in two terms,

$$\begin{aligned}
&\mathcal{L}^{\text{imp}}_{\text{est}}(\mathcal{T}, \Theta, D, T) \\
&= \mathcal{L}_{\text{est/w}}(\mathcal{T}, \mathcal{D}, \Theta, D) + \mathcal{L}_{\text{est}/\hat{\text{w}}}(\mathcal{T}, \Theta, D, T)
\end{aligned}$$

where $\mathcal{L}_{\text{est/w}}(\mathcal{T}, \mathcal{D}, \Theta, D)$ refers to the estimation error resulting from the finite size of D, assuming access to the true importance weights, and

$\mathcal{L}_{\text{est}/\hat{\text{w}}}(\mathcal{T}, \Theta, D, T)$ isolate the residual error resulting from the estimation of w. We have

$$\begin{aligned}&\mathcal{L}_{\text{est}/\text{w}}(\mathcal{T}, \mathcal{D}, \Theta, D)\\ &\qquad = \mathcal{L}(\theta_{\text{imp}(D,\mathcal{D})}; \mathcal{D}) - \min_\theta \mathcal{L}(\theta; \mathcal{T}),\\ &\mathcal{L}_{\text{est}/\hat{\text{w}}}(\mathcal{T}, \Theta, D, T)\\ &\qquad = \mathcal{L}(\theta_{\text{imp}(D,\mathcal{T})}; \mathcal{D}) - \mathcal{L}(\theta_{\text{imp}(D,T)}; \mathcal{D})\end{aligned}$$

with $\theta_{\text{imp}(D,\mathcal{D})} = \arg\min_\theta \mathcal{L}_{\text{imp}}(\theta; D, T, \hat{w})$

The first term depends on the size of D and the imbalance of the weights. For instance, if the weights are mostly concentrated over a small subset of D, this estimation error will be high. If this subset is smaller than T, estimation errors from $\mathcal{L}_{\text{imp}}(\theta; D, T, \hat{w})$ will be higher than from $\mathcal{L}(\theta; T)$. The notion of *effective sample size* has been defined to quantify this effect (Kish, 1965). It is defined by examining the variance of the weighted sum of n independent random variables Z_i with mean μ_Z and variance σ_Z^2, $S_w = \frac{\sum_i w_i Z_i}{\sum_i w_i}$. This variance is

$$\sigma_{S_w}^2 = \frac{\sum_i w_i^2}{(\sum_i w)^2}\sigma_Z^2$$

which can be compared to $\sigma_S^2 = \frac{1}{n}\sigma_Z^2$ in the unweighted case. This means that the weighted sum variance matches the variance of an unweighted case with

$$n_e = \frac{(\sum_i w)^2}{\sum_i w_i^2}.$$

Assuming that losses over $\mathcal{D}$ and $\mathcal{T}$ have comparable means and variances, the expected loss estimate with importance weighting over D has lower variance than the mean over T only when,

$$n_e = \frac{(\overline{w})^2}{\overline{w^2}}|D| \gg |T|$$

where $\overline{w} = \frac{1}{|D|}\sum_{y\in D} w(y)$ and $\overline{w^2} = \frac{1}{|D|}\sum_{y\in D} w^2(y)$ are the sample mean and variance of the weights over D. This means that the first term in the estimation error is $\mathcal{L}_{\text{est}/\text{w}}(\mathcal{T}, \Theta, D, T)$ advantageous compared to the estimation error from classical empirical risk minimization over T when T is small.

Unfortunately, the second estimation error term $\mathcal{L}_{\text{est}/\hat{\text{w}}}(\mathcal{T}, \Theta, D, T)$ gets larger as T gets smaller since estimating the importance weights $w(y; \mathcal{T}, \mathcal{D}) = \frac{P(y|\mathcal{T})}{P(y|\mathcal{D})}$ from data is challenging when T is small. One can remark that language modeling is actually the very problem of identifying a model to estimate the probabilities in this ratio, $P(y|\mathcal{T})$ and $P(y|\mathcal{D})$, from finite samples from the distributions $\mathcal{T}$ and $\mathcal{D}$. Discriminative classifiers are also relevant to estimate this ratio since

$$w(y; \mathcal{T}, \mathcal{D}) \propto \frac{P(\mathcal{T}|y)}{P(\mathcal{D}|y)}.$$

In fact the multiplying constant (prior ratio) does not matter since multiplying the weighted loss by a positive constant has no impact on optimization.

When importance weights are estimated with an LM, one can estimate $P(\cdot|\mathcal{T})$ by fine tuning on T a model pre-trained on D. The number of tuning steps n_{ft} gives controls on $\|\theta_{\text{ft}} - \theta_D\|$. When $n_{\text{ft}} = 0$, $\hat{w} = 1$ and the importance sampling loss corresponds to $\mathcal{L}(\theta, D)$. As n_{ft} grows, the estimate $P(y|\theta^{\text{ft}})$ could overfit and assigns most of the probability mass to a small neighborhood around samples in T. The weights $\hat{w}$ will in turn be concentrated in this small neighborhood, making the minimizer of the importance sampling loss close to the minimizer of the empirical loss over T. Therefore, fine-tuning a language model for estimating the importance weights allow to progressively transition between the in-domain and the out-of-domain empirical loss minimizers seen in Section 3.2. In the next sections, we refer to the estimated importance sampling weights as

$$w_{D,T}^{\text{imp}}(y) = \hat{w}(y; T, D).$$

Importance sampling has been used for model training for various application: either to improve training speed (Johnson and Guestrin, 2018; Katharopoulos and Fleuret, 2018) or to adapt to a changing training distribution (Mahmood et al., 2014; Metelli et al., 2018). Importance sampling has rarely been used to modify the training distribution of language models (Foster et al., 2010; Fernandez and Downey, 2018) as intelligent selection methods are more common.

4.2 Intelligent Selection

Intelligent selection (Moore and Lewis, 2010; Axelrod et al., 2011) and contrastive data selection, its extension to neural networks (van der Wees et al., 2017; Wang et al., 2018), have been introduced in the language modeling literature. We show that these methods are closely related to importance sampling, even if their original papers does not mention this link.

Intelligent selection selects training samples from an out-of-domain dataset according to the log-odd between an in-domain LM and an out-of-domain LM. Typically, a binary decision is taken per sentence by comparing the average log-odd to a threshold τ,

$$\mathcal{L}^{\text{IntSel}}(\theta, D, T) = -\sum_{y \in D} b^{\text{IntSel}}_{D,T}(y) \log P(y|\theta)$$

where $b^{\text{IntSel}}_{D,T}(y)$ is defined as $\text{I}\left\{\log P(y|\theta_T) - \log P(y|\theta_D) > \tau\right\}$. Compared to importance sampling, the weights are binarized, i.e.

$$\boxed{b^{\text{IntSel}}_{D,T}(y) = \text{I}\left\{\log w^{\text{imp}}_{D,T}(y) > \tau\right\}.}$$

The binarization decision was certainly driven by convenience as most n-gram LM training packages did not support weighted likelihood optimization when intelligent selection was introduced. Binarization also has the advantage of down-weighting extremely positive weight values from large $\log P(y|\theta_T)$ due to over-fitting on the small set T.

More recently, intelligent selection has been extended to neural models (van der Wees et al., 2017; Wang et al., 2018). Contrastive data selection (Wang et al., 2018) suggests to fine tune the in-domain model $\log P(y|\theta_T)$ from $\log P(y|\theta_D)$ and also observes that selection scores can efficiently be estimated from a model with a much smaller capacity than the final trained model. Dynamic selection (van der Wees et al., 2017) proposes to increase the selection threshold τ_t as training progresses, gradually transitioning from generic to in-domain training. This gradual adaptation of neural network is related to curriculum learning (Bengio et al., 2009) which studies the ordering of examples and tasks during model training.

Intelligent selection methods have been applied both for unconditional models (language modeling) and conditional models (machine translation). In the conditional case, intelligent selection computes

$$b^{\text{IntSel}}_{D,T}(x,y) = \text{I}\left\{\log w^{\text{IntSel}}_{D,T}(x,y) > \tau\right\}$$
$$\text{with} \quad w^{\text{IntSel}}_{D,T}(x,y) = \frac{P(y|x,\theta_T)}{P(y|x,\theta_D)}.$$

This ratio of conditional probabilities is different from the ratio of joint probabilities stemming from importance sampling, i.e.

$$\mathcal{L}_{\text{imp}}(\theta; D, T, \hat{w}) = - \frac{1}{|D|} \sum_{y \in D} \frac{P(x,y|\mathcal{T})}{P(x,y|\mathcal{D})} \log P(y|x,\theta).$$

The two ratios match when $P(x|\mathcal{T}) = P(x|\mathcal{D})$ since

$$\boxed{\begin{aligned} w^{\text{imp}}_{D,T}(x,y) &= \frac{P(x,y|\mathcal{T})}{P(x,y|\mathcal{D})} \\ &= \frac{P(x|\mathcal{T})}{P(x|\mathcal{D})} w^{\text{IntSel}}_{D,T}(x,y). \end{aligned}}$$

The formulation of intelligent selection therefore neglects the domain mismatch from the input distribution in the conditional case. This formulation aligns with the denoising goal (Wang et al., 2018) which assumes that D contains label noise, i.e. mistranslation in that case.

4.3 Influence Functions

As mentioned above, importance sampling and intelligent selection weights can be estimated by contrasting the log probabilities from a base model with those from a fine-tuned model. This use of fine-tuning connects intelligent selection to influence function and gradient alignment techniques. Influence functions (Koh and Liang, 2017; Pruthi et al., 2020) have been used as a diagnostic tool to identify the training instances which support or contradict with a given test label. This task is related to the selection of training data when the objective is to find instances in a generic training set D whose training updates increase the likelihood of a set T from a different domain.

The influence of a training point y on a test point y' is defined as

$$\text{I}(y, y') = -\frac{\partial \ell}{\partial \theta}(y'; \theta)^\top H_\theta^{-1} \frac{\partial \ell}{\partial \theta}(y; \theta)$$

where $\ell(y, \theta)$ refers to the loss at y for a model with parameters θ and H_θ refers to the Hessian of the model loss at θ. This quantity can be derived by considering the impact of reducing the weight of point y during training on the test loss at y'. If we increase the weight of a training example by ϵ,

$$\theta_{D,\epsilon} = \min_\theta \frac{1}{|D|} \sum_{z \in D} \ell(z; \theta) + \epsilon \ell(y; \theta)$$

From (Cook and Weisberg, 1982), we derive

$$\left.\frac{\partial \theta_{D,\epsilon}}{\partial \epsilon}\right|_{\epsilon=0} = -H_\theta^{-1} \left.\frac{\partial \ell}{\partial \theta}(y;\theta)\right|_{\theta=\theta_D}.$$

Composing with the test loss on (x', y'), we get

$$\left.\frac{\partial \ell(y';\theta_{D,\epsilon})}{\partial \epsilon}\right|_{\epsilon=0} = -\left.\frac{\partial \ell(y';\theta)^\top}{\partial \theta}\right|_{\theta=\theta_D} H_\theta^{-1} \left.\frac{\partial \ell(y;\theta)}{\partial \theta}\right|_{\theta=\theta_D}$$

which matches the expression of influence introduced above.

We now connect influence with the precedent sections on importance sampling and contrastive data selection. We consider an LM with weights θ_D, trained on the generic training set D. Its first order Taylor expansion at θ_D is

$$\begin{aligned}&\log P(y|\theta_D + \Delta\theta) = \\ &\log P(y|\theta_D) + \Delta\theta^\top g(y;\theta_D) + O\left(\|\Delta\theta\|^2\right) \quad (3)\end{aligned}$$

where $g(y;\theta_D) = \frac{\partial}{\partial\theta} \log P(y|\theta)\big|_{\theta=\theta_D}$. If the model pre-trained on D is fine-tuned on T by performing a single step of gradient descent with learning rate λ, we get

$$\begin{aligned}\theta_T &= \theta_D - \lambda \left.\frac{\partial}{\partial\theta}\mathcal{L}(T;\theta)\right|_{\theta=\theta_D} \\ &= \theta_D + \lambda \mathop{\mathbb{E}}_{y\sim T}\left[g(y;\theta_D)\right].\end{aligned}$$

In that case, the log-odd of the two models therefore has the following Taylor expansion,

$$\begin{aligned}&\log P(y|\theta_T) - \log P(y|\theta_D) \\ &\quad = \lambda \mathop{\mathbb{E}}_{y'\sim T}\left[g(y';\theta_D)^\top g(y;\theta_D)\right] \\ &\qquad\qquad + O\left(\|\theta_D - \theta_T\|^2\right).\end{aligned}$$

If we assume that the model's Hessian is the identity, $H_\theta = \mathbb{1}$, we therefore have

$$\begin{aligned}&\log P(y|\theta_T) - \log P(y|\theta_D) = \\ &\quad -\lambda \mathop{\mathbb{E}}_{y'\sim T}\left[\mathrm{I}(y,y')\right] + O\left(\|\theta_D - \theta_T\|^2\right).\end{aligned}$$

The Hessian assumption might be dropped when the model is fine-tuned with a Newton-style update (Boyd and Vandenberghe, 2014). The above relation means that the negative mean influence of a point $y \in D$ over the set T also corresponds to the log of the estimated importance weights introduced in Section 4.1, i.e.

$$\boxed{\begin{aligned}&\log w^{\mathrm{imp}}_{D,T}(y) = \\ &\quad -\lambda \mathop{\mathbb{E}}_{y'\sim T}\left[\mathrm{I}(y,y')\right] + O\left(\|\theta_D - \theta_T\|^2\right).\end{aligned}}$$

Of course, this relation holds only in the case where a single gradient step is performed for fine-tuning. This relation allows estimating the reduction in test loss (here over T) when removing training samples from D with positive influence which is also the goal of intelligent data selection. This strategy has been applied to label noise filtering (Koh and Liang, 2017), class rebalancing (Ren et al., 2018) and domain adaptation (Wang et al., 2021).

4.4 Comparing Data Selection Methods

Our analysis connects importance sampling, contrastive data selection and influence functions. In practice, contrastive data selection is the most popular approach. Unlike influence functions, contrastive data selection weights rely on fine tuning the generic model for more than one step on the in-domain data T. This has two effects. On one hand the contrastive data selection weights can be more reliable, closer to the ideal weights $w(y;\mathcal{T},\mathcal{D}) = \frac{P(y|\mathcal{T})}{P(y|\mathcal{D})}$. On the other hand, multiple steps increase the risk of over-fitting to T. In the case where one first trains with data selection before fine tuning on T, it might actually be helpful to limit the influence of T on selected data, to increase the complementary effect of fine tuning (Iter and Grangier, 2021).

When comparing contrastive data selection with importance sampling, the weight binarization is the main difference. This binarization might also have two opposite effects. On the positive side, it acts has a regularizer since binary weights are less likely to reflect statistics specific to T compared to unquantized ones. On the negative side, it cancels low weights which might collectively represent most of the weighted cross entropy. This interpretation of contrastive data selection as a regularized version of importance sampling opens the door to exploring more sophisticated regularization alternative to regularization, e.g. using a lower capacity model or different input features to estimate selection weights.

5 Conclusions

This work focuses on domain adaptation for neural language modeling. It compares the generalization properties of a model trained over a large out-of-domain corpus as opposed to a model trained over a small in-domain corpus. It shows how fine-tuning, the most common approach for neural LM adaptation can achieve better trade-offs than either solution. We then focus on adaptation via data selection techniques, i.e. techniques to emphasize in-domain data in an out-of-domain training set. We show that common techniques, contrastive data selection and influence function selection, can both be derived from importance sampling.

Our analysis currently assumes a pure language modeling setup, i.e. an auto-regressive model trained aiming high log-likelihood, both for out-of-domain and in-domain data. In the future, we want to extend our analysis of domain adaptation techniques to the popular setting (Bommasani et al., 2021) where model training combines language modeling over out-of-domain data and a different final task on in-domain data.

Our theoretical work also raises empirical questions. The binarization of importance sampling weights in intelligent selection is a simple variance reduction technique and more sophisticated alternative might be beneficial empirically. The link between influence functions and importance sampling suggests that examples with importance sampling weights lower than one have only a negative effect on the in-domain likelihood, which is not a typical observation in practice. This contradiction suggests expanding influence scores to take into account effects beyond a single update.

Acknowledgements

We thanks Wei Wang, Bowen Liang, Kelvin Guu and Nicolas Le Roux for their suggestions and comments.

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. *arXiv preprint arXiv:2004.02105*.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48. ACM.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. 2021. Foundation models. *CoRR*, abs/2108.07258.

Léon Bottou and Olivier Bousquet. 2007. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 161–168. Curran Associates, Inc.

Stephen P. Boyd and Lieven Vandenberghe. 2014. *Convex Optimization*. Cambridge University Press.

Rich Caruana. 1998. Multitask learning. In Sebastian Thrun and Lorien Y. Pratt, editors, *Learning to Learn*, pages 95–133. Springer.

Rich Caruana, Steve Lawrence, and Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Advances in neural information processing systems*, pages 402–408.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

R Dennis Cook and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jared Fernandez and Doug Downey. 2018. Sampling informative training data for RNN language models. In *Proceedings of ACL 2018, Student Research Workshop*, pages 9–13, Melbourne, Australia. Association for Computational Linguistics.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459, Cambridge, MA. Association for Computational Linguistics.

Ken-Ichi Funahashi. 1989. On the approximate realization of continuous mappings by neural networks. *Neural networks*, 2(3):183–192.

D. Grangier and S. Bengio. 2008. A discriminative kernel-based model to rank images from text queries. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *CoRR*, abs/2004.10964.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. *CoRR*, abs/1902.00751.

Dan Iter and David Grangier. 2021. On the complementarity of data selection and fine tuning for domain adaptation. *arXiv*, 2109.07591.

Tyler B Johnson and Carlos Guestrin. 2018. Training deep models faster with robust, approximate importance sampling. *Advances in Neural Information Processing Systems*, 31:7265–7275.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Angelos Katharopoulos and François Fleuret. 2018. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR.

Leslie Kish. 1965. Survey sampling. new york: John wesley & sons.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894, International Convention Centre, Sydney, Australia. PMLR.

Yann Lecun. 1987. *Modeles connexionnistes de l'apprentissage*. Universite Pierre et Marie Curie (Paris 6).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ashique Rupam Mahmood, Hado Van Hasselt, and Richard S Sutton. 2014. Weighted importance sampling for off-policy learning with linear function approximation. In *NIPS*, pages 3014–3022.

Christopher Manning and Hinrich Schutze. 1999. *Foundations of statistical natural language processing*. MIT press.

Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. 2018. Policy optimization via importance sampling. *arXiv preprint arXiv:1809.06098*.

Antonio Valerio Miceli Barone, Barry Haddow, Ulrich Germann, and Rico Sennrich. 2017. Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1494, Copenhagen, Denmark. Association for Computational Linguistics.

Tomás Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048. ISCA.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Art B. Owen. 2013. *Monte Carlo theory, methods and examples*. Stanford.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterfusion: Non-destructive task composition for transfer learning.

Jonathan Pilault, Amine El hattami, and Christopher Pal. 2021. Conditionally adaptive multi-task learning: Improving transfer learning in {nlp} using fewer parameters & less data. In *International Conference on Learning Representations*.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners (gpt).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pages 4334–4343. PMLR.

Szymon Roziewski and Marek Kozłowski. 2021. Languagecrawl: a generic tool for building language models upon common crawl. *Language Resources and Evaluation*, pages 1–29.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368.

Asa Cooper Stickland, Iain Murray, someone, and someone. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995, Long Beach, California, USA. PMLR.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *INTERSPEECH 2012, 13th Annual Conference of the International Speech Communication Association, Portland, Oregon, USA, September 9-13, 2012*, pages 194–197. ISCA.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 133–143, Brussels, Belgium. Association for Computational Linguistics.

Xinyi Wang, Ankur Bapna, Melvin Johnson, and Orhan Firat. 2021. Gradient-guided loss masking for neural machine translation.

A Proof of Theorem 1

When the distributions $\mathcal{T}$ and $\mathcal{D}$ differ, the Kullback–Leibler (KL) divergence between the two distributions is considered. We show that the generalization of the loss of θ_D over T is upper bounded

$$\forall \epsilon > 0,\ \exists N \text{ s.t. } \forall D \sim \mathcal{D}^n, \\ \mathcal{L}(\theta_D; \mathcal{T}) \leq H(\mathcal{T}) + KL(\mathcal{T}, \mathcal{D}) + \epsilon \quad (4)$$

with probability $1 - \epsilon$ This bound justifies intuition that if given the choice between two generic domain $\mathcal{D}$ and $\mathcal{D}'$, training over the one with the lowest KL divergence to $\mathcal{T}$ will result a in better asymptotic behaviour.

Proof. We consider the asymptotic case for the size of D. For any $\epsilon > 0$, the universal approximation property allows us to consider a model capacity large enough such that

$$\mathcal{L}_{\text{app}}(\mathcal{D}, \Theta) < \frac{\epsilon}{2}$$

Using consistency, we can also consider a training set D large enough such that

$$\mathcal{L}_{\text{est}}(\mathcal{D}, \Theta, D) < \frac{\epsilon}{2}$$

with probability $1 - \epsilon$. With the same probability,

$$\mathcal{L}(\theta_D; \mathcal{D}) < \mathcal{L}_H(\mathcal{D}) + \epsilon$$

which can be rewritten as a bound on the Kullback-Leibler divergence,

$$KL(P(\cdot|\mathcal{D}), P(\cdot|\theta_D)) = \mathcal{L}(\theta_D; \mathcal{D}) - \mathcal{L}_H(\mathcal{D}) < \epsilon.$$

This bound can help connecting the generalization loss of θ_D over T with the Kullback-Leibler divergence of T and D,

$$\begin{aligned} & \mathcal{L}(\theta_D; \mathcal{T}) \\ = & \sum_{y \in \Omega} P(y|\mathcal{T}) \log P(y|\theta_D) \\ = & \sum_{y \in \Omega} P(y|\mathcal{T}) \log(P(y|\mathcal{D}) + P(y|\theta_D) - P(y|\mathcal{D})) \\ \leq & \sum_{y \in \Omega} P(y|\mathcal{T}) \log(P(y|\mathcal{D}) + |P(y|\mathcal{D}) - P(y|\theta_D)|) \\ \leq & \sum_{y \in \Omega} P(y|\mathcal{T}) \log(P(y|\mathcal{D}) + 2\epsilon^2) \qquad (5) \\ \leq & \sum_{y \in \Omega} P(y|\mathcal{T}) \log(P(y|\mathcal{D})) + \log(1 + 2m\epsilon^2) \\ \leq & \ H(\mathcal{T}) + KL(\mathcal{T}, \mathcal{D}) + \log(1 + 2m\epsilon^2) \end{aligned}$$

where $m = 1/\min_y P(y|\mathcal{D})$ assumes that $P(\cdot|\mathcal{D})$ has full support, and (5) relies on Pinsker's inequality, i.e. $\max_y |P(y) - Q(y)| < 2KL(Q, Y)^2$. $\square$