Balancing Training for
Multilingual Neural Machine Translation

Xinyi Wang   **Yulia Tsvetkov**   **Graham Neubig**
Language Technology Institute, Carnegie Mellon University, Pittsburgh, PA 15213
{xinyiw1,ytsvetko,gneubig}@cs.cmu.edu

Abstract

When training multilingual machine translation (MT) models that can translate to/from multiple languages, we are faced with imbalanced training sets: some languages have much more training data than others. Standard practice is to up-sample less resourced languages to increase representation, and the degree of up-sampling has a large effect on the overall performance. In this paper, we propose a method that instead automatically learns how to weight training data through a data scorer that is optimized to maximize performance on all test languages. Experiments on two sets of languages under both one-to-many and many-to-one MT settings show our method not only consistently outperforms heuristic baselines in terms of average performance, but also offers flexible control over the performance of which languages are optimized.[1]

1 Introduction

Multilingual models are trained to process different languages in a single model, and have been applied to a wide variety of NLP tasks such as text classification (Klementiev et al., 2012; Chen et al., 2018a), syntactic analysis (Plank et al., 2016; Ammar et al., 2016), named-entity recognition (Xie et al., 2018; Wu and Dredze, 2019), and machine translation (MT) (Dong et al., 2015; Johnson et al., 2016). These models have two particularly concrete advantages over their monolingual counterparts. First, deploying a single multilingual model is much more resource efficient than deploying one model for each language under consideration (Arivazhagan et al., 2019; Aharoni et al., 2019). Second, multilingual training makes it possible to transfer knowledge from high-resource languages (HRLs) to improve performance on low-resource languages (LRLs) (Zoph et al., 2016;

Nguyen and Chiang, 2018; Neubig and Hu, 2018; Wang and Neubig, 2019; Aharoni et al., 2019).

A common problem with multilingual training is that the data from different languages are both heterogeneous (different languages may exhibit very different properties) and imbalanced (there may be wildly varying amounts of training data for each language). Thus, while LRLs will often benefit from transfer from other languages, for languages where sufficient monolingual data exists, performance will often *decrease* due to interference from the heterogeneous nature of the data. This is especially the case for modestly-sized models that are conducive to efficient deployment (Arivazhagan et al., 2019; Conneau et al., 2019).

To balance the performance on different languages, the standard practice is to heuristically adjust the distribution of data used in training, specifically by over-sampling the training data from LRLs (Johnson et al., 2016; Neubig and Hu, 2018; Arivazhagan et al., 2019; Conneau et al., 2019). For example, Arivazhagan et al. (2019) sample training data from different languages based on the dataset size scaled by a heuristically tuned temperature term. However, such heuristics are far from perfect. First, Arivazhagan et al. (2019) find that the exact value of this temperature term significantly affects results, and we further show in experiments that the ideal temperature varies significantly from one experimental setting to another. Second, this heuristic ignores factors other than data size that affect the interaction between different languages, despite the fact that language similarity has been empirically proven important in examinations of cross-lingual transfer learning (Wang and Neubig, 2019; Lin et al., 2019).

In this paper, we ask the question: "is it possible to *learn* an optimal strategy to automatically balance the usage of data in multilingual model training?" To this effect, we propose a method that

---

[1]The code is available at https://github.com/cindyxinyiwang/fairseq/tree/multiDDS.

learns a language scorer that can be used throughout training to improve the model performance on *all* languages. Our method is based on the recently proposed approach of Differentiable Data Selection (Wang et al., 2019b, DDS), a general machine learning method for optimizing the weighting of different training examples to improve a pre-determined objective. In this work, we take this objective to be the average loss from different languages, and directly optimize the weights of training data from each language to maximize this objective on a multilingual development set. This formulation has no heuristic temperatures, and enables the language scorer to consider the interaction between languages.

Based on this formulation, we propose an algorithm that improves the ability of DDS to optimize *multiple* model objectives, which we name Multi-DDS. This is particularly useful in the case where we want to optimize performance on multiple languages simultaneously. Specifically, MultiDDS (1) has a more flexible scorer parameterization, (2) is memory efficient when training on multiple languages, and (3) stabilizes the reward signal so that it improves all objectives simultaneously instead of being overwhelmed by a single objective.

While the proposed methods are model-agnostic and thus potentially applicable to a wide variety of tasks, we specifically test them on the problem of training multilingual NMT systems that can translate many languages in a single model. We perform experiments on two sets of languages (one with more similarity between the languages, one with less) and two translation directions (one-to-many and many-to-one where the "one" is English). Results show that MultiDDS consistently outperforms various baselines in all settings. Moreover, we demonstrate MultiDDS provides a flexible framework that allows the user to define a variety of optimization objectives for multilingual models.

2 Multilingual Training Preliminaries

Monolingual Training Objective A standard NMT model is trained to translate from a single source language S to a target language T. The parameters of the model are generally trained by preparing a training dataset D_{train}, and defining the empirical distribution of sentence pairs $\langle x, y \rangle$ sampled from D_{train} as P. We then minimize the empirical risk $J(\theta, P)$, which is the expected value

of the loss function $\ell(x, y; \theta)$ over this distribution:

$$\theta^* = \underset{\theta}{\arg\min} \, J(\theta, D_{\text{train}})$$
$$\text{where} \quad J(\theta, D_{\text{train}}) = \mathbb{E}_{x,y \sim P(X,Y)}[\ell(x, y; \theta)] \tag{1}$$

Multilingual Training Formulation A multilingual NMT model can translate n pairs of languages $\{S^1\text{-}T^1, S^2\text{-}T^2, ..., S^n\text{-}T^n\}$, from any source language S^i. to its corresponding target T^i. To train such a multilingual model, we have access to n sets of training data $D_{\text{train}} = D_{\text{train}}^1, D_{\text{train}}^2, \dots, D_{\text{train}}^n$, where D_{train}^i is training data for language pair $S^i\text{-}T^i$. From these datasets, we can define P^i, the distribution of sentences from $S^i\text{-}T^i$, and consequently also define a risk $J(\theta, P^i)$ for each language following the monolingual objective in Eq. 1.

However, the question now becomes: "how do we define an overall training objective given these multiple separate datasets?" Several different methods to do so have been proposed in the past. To discuss all of these different methods in a unified framework, we further define a distribution P_D over the n sets of training data, and define our overall multilingual training objective as

$$J_{\text{mult}}(\theta, P_D, D_{\text{train}}) = \mathbb{E}_{i \sim P_D(i;\psi)} \left[J(\theta, D_{\text{train}}^i) \right]. \tag{2}$$

In practice, this overall objective can be approximated by selecting a language according to $\tilde{i} \sim P_D(i)$, then calculating gradients with respect to θ on a batch of data from $D_{\text{train}}^{\tilde{i}}$.

Evaluation Methods Another important question is how to evaluate the performance of such multilingual models. During training, it is common to use a separate development set for each language $D_{\text{dev}} = D_{\text{dev}}^1, D_{\text{dev}}^2, ..., D_{\text{dev}}^n$ to select the best model. Given that the objective of multilingual training is generally to optimize the performance on all languages simultaneously (Arivazhagan et al., 2019; Conneau et al., 2019), we can formalize this objective as minimizing the average of dev risks[2]:

$$J_{\text{dev}}(\theta, D_{\text{dev}}) = \frac{1}{n} \sum_{i=1}^{n} J(\theta, D_{\text{dev}}^i). \tag{3}$$

---

[2]In reality, it is common to have the loss ℓ be a likelihood-based objective, but finally measure another metric such as BLEU score at test time, but for simplicity we will assume that these two metrics are correlated.

Relation to Heuristic Strategies This formulation generalizes a variety of existing techniques that define $P_D(i)$ using a heuristic strategy, and keep it fixed throughout training.

Uniform: The simplest strategy sets $P_D(i)$ to a uniform distribution, sampling minibatches from each language with equal frequency (Johnson et al., 2016).

Proportional: It is also common to sample data in portions equivalent to the size of the corresponding corpora in each language (Johnson et al., 2016; Neubig and Hu, 2018).

Temperature-based: Finally, because both of the strategies above are extreme (proportional underweighting LRLs, and uniform causing overfitting by re-sampling sentences from limited-size LRL datasets), it is common to sample according to data size exponentiated by a temperature term τ (Arivazhagan et al., 2019; Conneau et al., 2019):

$$P_D(i) = \frac{q_i^{1/\tau}}{\sum_{k=1}^n q_k^{1/\tau}} \text{ where } q_i = \frac{|D_{\text{train}}^i|}{\sum_{k=1}^n |D_{\text{train}}^k|}. \quad (4)$$

When $\tau = 1$ or $\tau = \infty$ this is equivalent to proportional or uniform sampling respectively, and when a number in the middle is chosen it becomes possible to balance between the two strategies.

As noted in the introduction, these heuristic strategies have several drawbacks regarding sensitivity to the τ hyperparameter, and lack of consideration of similarity between the languages. In the following sections we will propose methods to resolve these issues.

3 Differentiable Data Selection

Now we turn to the question: is there a better way to optimize $P_D(i)$ so that we can achieve our final objective of performing well on a representative development set over all languages, i.e. minimizing $J_{\text{dev}}(\theta, D_{\text{dev}})$. In order to do so, we turn to a recently proposed method of Differentiable Data Selection (Wang et al., 2019b, DDS), a general purpose machine learning method that allows for weighting of training data to improve performance on a separate set of held-out data.

Specifically, DDS uses a technique called *bilevel optimization* (Colson et al., 2007), that learns

a second set of parameters ψ that modify the training objective that we use to learn θ, so as to maximize the final objective $J_{\text{dev}}(\theta, D_{\text{dev}})$. Specifically, it proposes to learn a data scorer $P(x, y; \psi)$, parameterized by ψ, such that training using data sampled from the scorer optimizes the model performance on the dev set. To take the example of learning an NMT system to translate a *single* language pair i using DDS, the general objective in Eq. 1 could be rewritten as

$$\psi^* = \underset{\psi}{\operatorname{argmin}} \, J(\theta^*(\psi), D_{\text{dev}}^i) \text{ where}$$
$$\theta^*(\psi) = \underset{\theta}{\operatorname{argmin}} \, \mathbb{E}_{x,y \sim P(x,y;\psi)} \left[\ell(x, y; \theta) \right]. \quad (5)$$

DDS optimizes θ and ψ iteratively throughout the training process. Given a fixed ψ, the update rule for θ is simply

$$\theta_t \leftarrow \theta_{t-1} - \nabla_\theta \mathbb{E}_{x,y \sim P(x,y;\psi)} \left[\ell(x, y; \theta) \right]$$

To update the data scorer, DDS uses reinforcement learning with a reward function that approximates the effect of the training data on the model's dev performance

$$R(x, y; \theta_t) \approx \nabla J(\theta_t, D_{\text{dev}}^i)^\top \cdot \nabla_\theta \ell(x, y; \theta_{t-1})$$
$$\approx \cos \left(\nabla J(\theta_t, D_{\text{dev}}^i), \nabla_\theta \ell(x, y; \theta_{t-1}) \right) \quad (6)$$

where $\cos(\cdot)$ is the cosine similarity of two vectors. This reward can be derived by directly differentiating $J(\theta(\psi), D_{\text{dev}}^i)$ with respect to ψ, but intuitively, it indicates that the data scorer should be updated to up-weigh the data points that have similar gradient with the dev data. According to the REINFORCE algorithm (Williams, 1992), the update rule for the data scorer then becomes

$$\psi_{t+1} \leftarrow \psi_t + R(x, y; \theta_t) \cdot \nabla_\psi \log P(x, y; \psi) \quad (7)$$

4 DDS for Multilingual Training

In this section, we use the previously described DDS method to derive a new framework that, instead of relying on fixed heuristics, adaptively optimizes usage of multilingual data for the best model performance on *multiple* languages. We illustrate the overall workflow in Fig. 1.

First, we note two desiderata for our multilingual training method: 1) **generality**: the method should be flexible enough so that it can be utilized universally for different multilingual tasks and settings (such as different translation directions for

NMT). 2) **scalablity**: the method should be stable and efficient if one wishes to scale up the number of languages that a multilingual model supports. Based on these two properties, we introduce Multi-DDS, an extension of the DDS method tailored for multilingual training.

Method MultiDDS directly parameterizes the standard dataset sampling distribution for multi-lingual training with ψ:

$$P_D(i;\psi) = e^{\psi_i} / \sum_{k=1}^n e^{\psi_k} \qquad (8)$$

and optimizes ψ to minimize the dev loss. Notably, unlike standard DDS we make the design decision to weight training datasets rather than score each training example $\langle x, y \rangle$ directly, as it is more efficient and also likely easier to learn.

We can thus rewrite the objective in Eq. 2 to incorporate both ψ and θ as:

$$
\begin{aligned}
\psi^* &= \operatorname*{argmin}_{\psi} J_{\text{dev}}(\theta^*(\psi), D_{\text{dev}}) \text{ where} \\
\theta^* &= \operatorname*{argmin}_{\theta} \mathbb{E}_{i \sim P_D(i;\psi)} \left[J(\theta, D_{\text{train}}^i) \right]
\end{aligned}
\qquad (9)
$$

In other words, while the general DDS framework evaluates the model performance on a single dev set and optimizes the weighting of each training example, our multilingual training objective evaluates the performance over an aggregation of n dev sets and optimizes the weighting of n training sets.

The reward signal for updating ψ_t is

$$
\begin{aligned}
R(i;\theta_t) &\approx \cos \left(\nabla \left(J_{\text{dev}}(\theta_t, D_{\text{dev}}) \right), \nabla_\theta J(\theta_{t-1}, D_{\text{train}}^i) \right) \\
&= \cos \left(\nabla \left(\frac{1}{n} \sum_{k=1}^n J(\theta_t, D_{\text{dev}}^k) \right), \nabla_\theta J(\theta_{t-1}, D_{\text{train}}^i) \right),
\end{aligned}
\qquad (10)
$$

where $J_{\text{dev}}(\cdot)$ defines the combination of n dev sets, and we simply plug in its definition from Eq. 3. Intuitively, Eq. 10 implies that we should favor the training language i if its gradient aligns with the gradient of the aggregated dev risk of all languages.

Implementing the Scorer Update The pseudo-code for the training algorithm using MultiDDS can be found in line 25. Notably, we do not update the data scorer ψ on every training step, because it is too computationally expensive for NMT training (Wang et al., 2019b). Instead, after training the multilingual model θ for a certain number of steps, we update the scorer for all languages. This implementation is not only efficient, but also allows us to
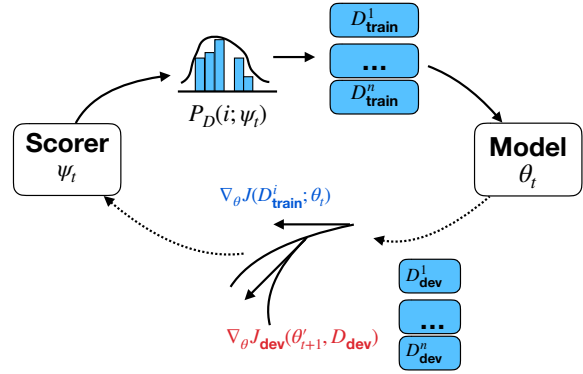

Figure 1: An illustration of the MultiDDS algorithm. Solid lines represent updates for θ, and dashed lines represent updates for ψ. The scorer defines the distribution over n training languages, from which training data is sampled to train the model. The scorer is updated to favor the datasets with similar gradients as the gradient of the aggregated dev sets.

re-estimate more frequently the effect of languages that have low probability of being sampled.

In order to do so, it is necessary to calculate the effect of each training language on the current model, namely $R(i; \theta_t)$. We estimate this value by sampling a batch of data from each D_{train}^i to get the training gradient for θ_t, and use this to calculate the reward for this language. This process is detailed in line 11 of the line 25.

Unlike the algorithm in DDS which requires storing n model gradients,[3] this approximation does not require extra memory even if n is large, which is important given recent efforts to scale multilingual training to 100+ (Arivazhagan et al., 2019; Aharoni et al., 2019) or even 1000+ languages (Östling and Tiedemann, 2017; Malaviya et al., 2017).

5 Stabilized Multi-objective Training

In our initial attempts to scale DDS to highly multi-lingual training, we found that one challenge was that the reward for updating the scorer became unstable. This is because the gradient of a multilingual dev set is less consistent and of higher variance than that of a monolingual dev set, which influences the fidelity of the data scorer reward. [4]

---

[3]The NMT algorithm in (Wang et al., 2019b) estimates the reward by storing the moving average of n training gradients, which is not memory efficient (See Line. 7 of Alg. 2 in (Wang et al., 2019b)). In the preliminary experiments, our approximation performs as well as the moving average approximation (see App. A.1). Thus, we use our approximation method as the component for MultiDDS for the rest of the experiments.

[4]Suppose the dev set gradient of language k has variance of $\text{var}(g_{\text{dev}}^k) = \sigma$, and that the dev gradients of each language

Algorithm 1: Training with MultiDDS

Input : $\mathcal{D}_{\text{train}}$; M: amount of data to train the multilingual model before updating ψ;

Output : The converged multilingual model θ^*

▷ *Initialize $P_D(i, \psi)$ to be proportional to dataset size*

1   $P_D(i, \psi) \leftarrow \frac{|D_{\text{train}}^i|}{\sum_{j=1}^n |D_{\text{train}}^j|}$

2   **while** *not converged* **do**

    ▷ *Load training data with ψ*

3     $X, Y \leftarrow \emptyset$

4   **while** $|X, Y| < M$ **do**

5       $\tilde{i} \sim P_D(i, \psi_t)$

6       $(x, y) \sim D_{\text{train}}^{\tilde{i}}$

7       $X, Y \leftarrow X, Y \cup x, y$

8   **end**

    ▷ *Train the NMT model for multiple steps*

9   **for** x, y *in* X, Y **do**

10       $\theta \leftarrow$ GradientUpdate $(\theta, \nabla_\theta \ell(x, y; \theta))$

11   **end**

    ▷ *Estimate the effect of each language $R(i; \theta)$*

12   **for** *i from 1 to n* **do**

13       $x', y' \sim D_{\text{train}}^i$

14       $g_{\text{train}} \leftarrow \nabla_\theta \ell(x', y'; \theta)$

15       $\theta' \leftarrow$ GradientUpdate $(\theta, g_{\text{train}})$

16       $g_{\text{dev}} \leftarrow 0$

17   **for** *j from 1 to n* **do**

18         $x_d, y_d \sim D_{\text{dev}}^j$

19         $g_{\text{dev}} \leftarrow g_{\text{dev}} + \nabla_{\theta'} \ell(x_d, y_d; \theta')$

20   **end**

21       $R(i; \theta) \leftarrow \cos(g_{\text{dev}}, g_{\text{train}})$

22   **end**

    ▷ *Optimize ψ*

23     $d_\psi \leftarrow \sum_{i=1}^n R(i; \theta) \cdot \nabla_\psi \log (P_D(i; \psi))$

24     $\psi \leftarrow$ GradientUpdate (ψ, d_ψ)

25   **end**

Thus, instead of using the gradient alignment between the training data and the aggregated loss of n dev sets as the reward, we propose a second approach to first calculate the gradient alignment reward between the data and each of the n dev sets, then take the average of these as the final reward.

$\{g_{\text{dev}}^1, ..., g_{\text{dev}}^n\}$ are independent. Then the sum of the gradients from the n languages has a variance of $\text{var}(\sum_{k=1}^n g_{\text{dev}}^k) = n\sigma$.

This can be expressed mathematically as follows:

$$R'(i; \theta_t) \approx$$
$$\cos\left(\nabla_\theta \left(\frac{1}{n} \sum_{k=1}^n J(\theta_t, D_{\text{dev}}^k)\right), \nabla_\theta J(\theta_{t-1}, D_{\text{train}}^i)\right)$$
$$\approx \frac{1}{n} \sum_{k=1}^n \cos\left(\nabla_\theta J(\theta_t, D_{\text{dev}}^k), \nabla_\theta J(\theta_{t-1}, D_{\text{train}}^i)\right)$$
$$(11)$$

To implement this, we can simply replace the standard reward calculation at Line 11 of line 25 to use the stable reward. We name this setting MultiDDS-S. In § 6.6 we show that this method has less variance than the reward in Eq. 10.

6 Experimental Evaluation

6.1 Data and Settings

We use the 58-languages-to-English parallel data from Qi et al. (2018). A multilingual NMT model is trained for each of the two sets of language pairs with different level of language diversity:

Related: 4 LRLs (Azerbaijani: `aze`, Belarusian: `bel`, Glacian: `glg`, Slovak: `slk`) and a related HRL for each LRL (Turkish: `tur`, Russian: `rus`, Portuguese: `por`, Czech: `ces`)

Diverse: 8 languages with varying amounts of data, picked without consideration for relatedness (Bosnian: `bos`, Marathi: `mar`, Hindi: `hin`, Macedonian: `mkd`, Greek: `ell`, Bulgarian: `bul`, French: `fra`, Korean: `kor`)

Statistics of the datasets are in § A.3.

For each set of languages, we test two varieties of translation: 1) many-to-one (M2O): translating 8 languages to English; 2) one-to-many (O2M): translating English into 8 different languages. A target language tag is added to the source sentences for the O2M setting (Johnson et al., 2016).

6.2 Experiment Setup

All translation models use standard transformer models (Vaswani et al., 2017) as implemented in fairseq (Ott et al., 2019) with 6 layers and 4 attention heads. All models are trained for 40 epochs. We preprocess the data using sentencpiece (Kudo and Richardson, 2018) with a vocabulary size of $8K$ for each language. The complete set of hyperparameters can be found in § A.2. The model performance is evaluated with BLEU score (Papineni et al., 2002), using sacreBLEU (Post, 2018).

Baselines We compare with the three standard heuristic methods explained in § 2: 1) Uniform ($\tau = \infty$): datasets are sampled uniformly, so that LRLs are over-sampled to match the size of the HRLs; 2) Temperature: scales the proportional distribution by $\tau = 5$ (following Arivazhagan et al. (2019)) to slightly over-sample the LRLs; 3) Proportional ($\tau = 1$): datasets are sampled proportional to their size, so that there is no over-sampling of the LRLs.

Ours   we run MultiDDS with either the standard reward (MultiDDS), or the stabilized reward proposed in Eq. 11 (MultiDDS-S). The scorer for MultiDDS simply maps the ID of each dataset to its corresponding probability (See Eq. 8. The scorer has N parameters for a dataset with N languages.)

6.3 Main Results

We first show the average BLEU score over all languages for each translation setting in Tab. 2. First, comparing the baselines, we can see that there is no consistently strong strategy for setting the sampling ratio, with proportional sampling being best in the M2O setting, but worst in the O2M setting. Next, we can see that MultiDDS outperforms the best baseline in three of the four settings and is comparable to proportional sampling in the last M2O-Diverse setting. With the stabilized reward, MultiDDS-S consistently delivers better overall performance than the best baseline, and outperforms MultiDDS in three settings. From these results, we can conclude that MultiDDS-S provides a stable strategy to train multilingual systems over a variety of settings.

Next, we look closer at the BLEU score of each language pair for MultiDDS-S and the best baseline. The results for all translation settings are in Tab. 1. In general, MultiDDS-S outperforms the baseline on more languages. In the best case, for the O2M-Related setting, MultiDDS-S brings significant gains for five of the eight languages, without hurting the remaining three. The gains for the Related group are larger than for the Diverse group, likely because MultiDDS can take better advantage of language similarities than the baseline methods.

It is worth noting that MultiDDS does not impose large training overhead. For example, for our M2O system, the standard method needs around 19 hours and MultiDDS needs around 20 hours for convergence. The change in training time is not siginificant because MultiDDS only optimizes a simple distribution over the training datasets.

6.4 Prioritizing what to Optimize

Prior works on multilingual models generally focus on improving the average performance of the model on all supported languages (Arivazhagan et al., 2019; Conneau et al., 2019). The formulation of MultiDDS reflects this objective by defining the aggregation of n dev sets using Eq. 3, which is simply the average of dev risks. However, average performance might not be the most desirable objective under all practical usage settings. For example, it may be desirable to create a more *egalitarian* system that performs well on all languages, or a more *specialized* system that does particularly well on a subset of languages.

In this section, we examine the possibility of using MultiDDS to control the priorities of the multilingual model by defining different dev set aggregation methods that reflect these priorities. To do so, we first train the model for 10 epochs using regular MultiDDS, then switch to a different dev set aggregation method. Specifically, we compare MultiDDS with three different priorities:

Regular: this is the standard MultiDDS that optimizes all languages throughout training using the average dev risk aggregation in Eq. 3

Low: a more egalitarian system that optimizes the average of the four languages with the worst dev perplexity, so that MultiDDS can focus on optimizing the low-performing languages

High: a more specialized system that optimizes the four languages with the best dev perplexity, for MultiDDS to focus on optimizing the high-performing languages

We performed experiments with these aggregation methods on the Diverse group, mainly because there is more performance trade-off among these languages. First, in Tab. 3 we show the average BLEU over all languages, and find that MultiDDS with different optimization priorities still maintains competitive average performance compared to the baseline. More interestingly, in Fig. 2, we plot the BLEU score difference of High and Low compared to Regular for all 8 languages. The languages are ordered on the x-axis from left to right in decreasing perplexity. Low generally performs better on the low-performing languages on the left, while High generally achieves the best performance on the high-performing languages on

	Method	Avg.	aze	bel	glg	slk	tur	rus	por	ces
M2O	Prop.	24.88	11.20	17.17	27.51	28.85	**23.09***	**22.89**	**41.60**	26.80
	MultiDDS-S	**25.52**	**12.20***	**19.11***	**29.37***	**29.35***	22.81	22.78	41.55	**27.03**
O2M	Temp.	16.61	**6.66**	11.29	**21.81**	18.60	11.27	14.92	32.10	16.26
	MultiDDS-S	**17.32**	6.59	**12.39***	21.65	**20.61***	**11.58**	**15.26***	**33.52***	**16.98***
			bos	mar	hin	mkd	ell	bul	fra	kor
M2O	Prop.	26.68	23.43	10.10	22.01	31.06	**35.62***	**36.41***	**37.91***	**16.91**
	MultiDDS-S	**27.00**	**25.34***	**10.57**	**22.93***	**32.05***	35.27	35.77	37.30	16.81
O2M	Temp.	17.94	**14.73***	**4.93**	15.49	20.59	24.82	26.60	**29.74***	6.62
	MultiDDS-S	**18.24**	14.02	4.76	**15.68***	**21.44**	**25.69***	**27.78***	29.60	**7.01***

Table 1: BLEU scores of the best baseline and MultiDDS-S for all translation settings. MultiDDS-S performs better on more languages. For each setting, bold indicates the highest value, and ∗ means the gains are statistically significant with $p < 0.05$.

	Method	M2O		O2M	
		Related	Diverse	Related	Diverse
Baseline	Uni. ($\tau=\infty$)	22.63	24.81	15.54	16.86
	Temp. ($\tau=5$)	24.00	26.01	16.61	17.94
	Prop. ($\tau=1$)	24.88	26.68	15.49	16.79
Ours	MultiDDS	25.26	26.65	17.17	**18.40**
	MultiDDS-S	**25.52**	**27.00**	**17.32**	18.24

Table 2: Average BLEU for the baselines and our methods. Bold indicates the highest value.

Setting	Baseline	MultiDDS-S		
		Regular	Low	High
M2O	26.68	27.00	26.97	27.08
O2M	17.94	18.24	17.95	18.55

Table 3: Average BLEU of the best baseline and three MultiDDS-S settings for the Diverse group. MultiDDS-S always outperform the baseline.
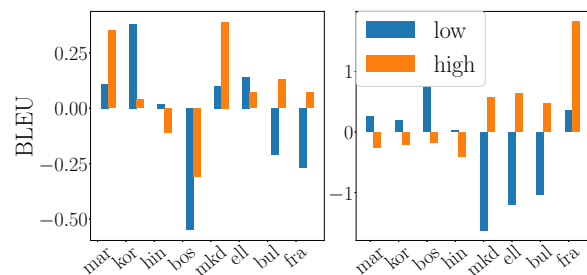

Figure 2: The difference between Low and High optimization objectives compared to Regular for the Diverse language group. MultiDDS successfully optimize for different priorities. *left*: M2O; *right*: O2M.

the right, with results most consistent in the O2M setting. This indicates that MultiDDS is able to prioritize different predefined objectives.

It is also worth noting that low-performing languages are not always low-resource languages. For example, Korean (kor) has the largest amount of training data, but its BLEU score is among the lowest. This is because it is typologically very different from English and the other training languages. Fig. 2 shows that Low is still able to focus on improving kor, which aligns with the predefined objective. This fact is not considered in baseline methods that only consider data size when sampling from the training datasets.

6.5 Learned Language Distributions

In Fig. 3, we visualize the language distribution learned by MultiDDS throughout the training process. Under all settings, MultiDDS gradually increases the usage of LRLs. Although initialized

with the same distribution for both one-to-many and many-to-one settings, MultiDDS learns to up-sample the LRLs more in the one-to-many setting, likely due to the increased importance of learning language-specific decoders in this setting. For the Diverse group, MultiDDS learns to decrease the usage of Korean (kor) the most, probably because it is very different from other languages in the group.

6.6 Effect of Stablized Rewards

Next, we study the effect of the stablized reward proposed in § 2. In Fig. 4, we plot the regular reward (used by MultiDDS) and the stable reward (used by MultiDDS-S) throughout training. For all settings, the reward in MultiDDS and MultiDDS-S follows the similar trend, while the stable reward used in MultiDDS-S has consistently less variance.

MultiDDS-S also results in smaller variance in the final model performance. We run Multi-DDS and MultiDDS-S with 4 different random seeds, and record the mean and variance of the average BLEU score. Tab. 4 shows results for the Diverse group, which indicate that the model performance achieved using MultiDDS-S has lower
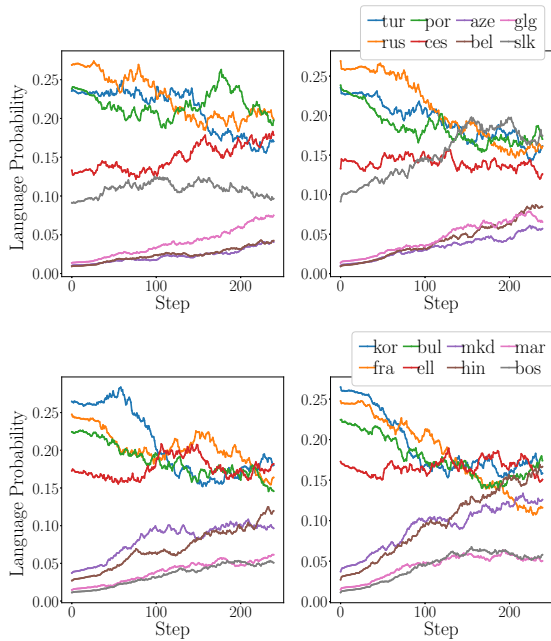
Figure 3: Language usage by training step. *Left*: many-to-one; *Right*: one-to-many; *Top*: related language group; *Bottom*: diverse language group.

Method	M2O		O2M	
	Mean	**Var.**	**Mean**	**Var.**
MultiDDS	26.85	0.04	18.20	0.05
MultiDDS-S	26.94	0.02	18.24	0.02

Table 4: Mean and variance of the average BLEU score for the Diverse group. The models trained with MultiDDS-S perform better and have less variance.

variance and a higher mean than MultiDDS.

Additionally, we compare the learned language distribution of MultiDDS-S and MultiDDS in Fig. 5. The learned language distribution in both plots fluctuates similarly, but MultiDDS has more drastic changes than MultiDDS-S. This is also likely due to the reward of MultiDDS-S having less variance than that of MultiDDS.

7 Related Work

Our work is related to the multilingual training methods in general. Multilingual training has a rich history (Schultz and Waibel, 1998; Mimno et al., 2009; Shi et al., 2010; Täckström et al., 2013), but has become particularly prominent in recent years due the ability of neural networks to easily perform multi-task learning (Dong et al., 2015; Plank et al., 2016; Johnson et al., 2016). As stated previously, recent results have demonstrated the importance of balancing HRLs and LRLs during multilingual training (Arivazhagan et al., 2019; Conneau et al.,
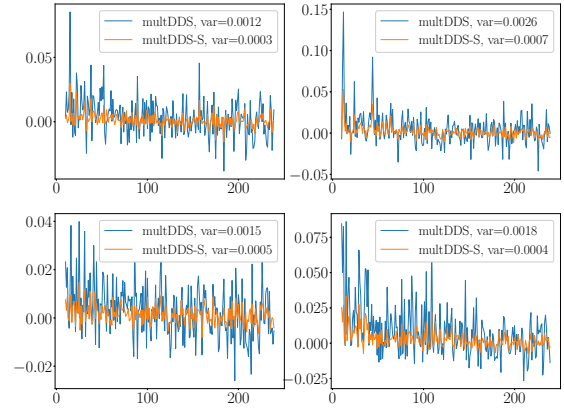

Figure 4: Variance of reward. *Left*: M2O; *Right*: O2M; *Top*: Related language group; *Bottom*: Diverse language group.
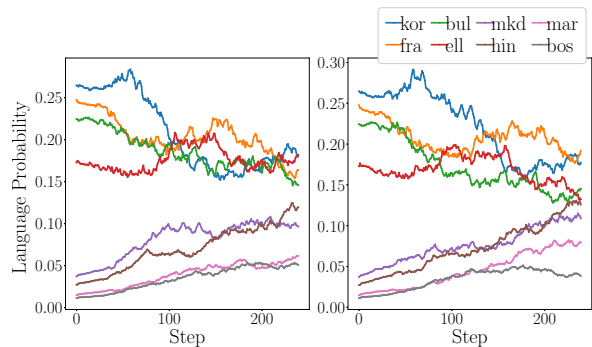

Figure 5: Language usage for the M2O-Diverse setting. *Left*: MultiDDS-S; *Right*: MultiDDS. The two figures follow similar trends while MultiDDS changes more drastically.

2019), which is largely done with heuristic sampling using a temperature term; MultiDDS provides a more effective and less heuristic method. Wang and Neubig (2019); Lin et al. (2019) choose languages from multilingual data to improve the performance on a particular language, while our work instead aims to train a single model that handles translation between many languages. (Zaremoodi et al., 2018; Wang et al., 2018, 2019a) propose improvements to the model architecture to improve multilingual performance, while MultiDDS is a model-agnostic and optimizes multilingual data usage.

Our work is also related to machine learning methods that balance multitask learning (Chen et al., 2018b; Kendall et al., 2018). For example, Kendall et al. (2018) proposes to weigh the training loss from a multitask model based on the uncertainty of each task. Our method focuses on optimizing the multilingual data usage, and is both somewhat orthogonal to and less heuristic than such loss weighting methods. Finally, our work is related to meta-learning, which is used in hyperpa-

rameter optimization (Baydin et al., 2018), model initialization for fast adaptation (Finn et al., 2017), and data weighting (Ren et al., 2018). Notably, Gu et al. (2018) apply meta-learning to learn an NMT model initialization for a set of languages, so that it can be quickly fine-tuned for any language. This is different in motivation from our method because it requires an adapted model for each of the language, while our method aims to optimize a single model to support all languages. To our knowledge, our work is the first to apply meta-learning to optimize data usage for multilingual objectives.

8 Conclusion

In this paper, we propose MultiDDS, an algorithm that learns a language scorer to optimize multilingual data usage to achieve good performance on many different languages. We extend and improve over previous work on DDS (Wang et al., 2019b), with a more efficient algorithmic instantiation tailored for the multilingual training problem and a stable reward to optimize multiple objectives. MultiDDS not only outperforms prior methods in terms of overall performance on all languages, but also provides a flexible framework to prioritize different multilingual objectives.

Notably, MultiDDS is not limited to NMT, and future work may consider applications to other multilingual tasks. In addition, there are other conceivable multilingual optimization objectives than those we explored in § 6.4.

Acknowledgement

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *NAACL*.

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2016. Many languages, one parser. *TACL*, 4:431–444.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. In *arxiv*.

Atilim Gunes Baydin, Robert Cornish, David Martínez-Rubio, Mark Schmidt, and Frank Wood. 2018. Online learning rate adaptation with hypergradient descent. In *ICLR*.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018a. Adversarial deep averaging networks for cross-lingual sentiment classification. In *ACL*.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018b. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*.

Benoît Colson, Patrice Marcotte, and Gilles Savard. 2007. An overview of bilevel optimization. *Annals OR*, 153(1).

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. In *EMNLP*.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *ACL*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *EMNLP*.

Melvin Johnson et al. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation. In *TACL*.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. In *Proceedings of COLING 2012*, pages 1459–1474.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *EMNLP*.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In *ACL*.

Chaitanya Malaviya, Graham Neubig, and Patrick Littell. 2017. Learning language representations for typology prediction. In *EMNLP*.

David M. Mimno, Hanna M. Wallach, Jason Naradowsky, David A. Smith, and Andrew McCallum. 2009. Polylingual topic models. In *EMNLP*.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. *EMNLP*.

Toan Q. Nguyen and David Chiang. 2018. Transfer learning across low-resource, related languages for neural machine translation. In *NAACL*.

Toan Q. Nguyen and Julián Salazar. 2019. Transformers without tears: Improving the normalization of self-attention. In *IWSLT*.

Robert Östling and Jörg Tiedemann. 2017. Continuous multilinguality with language vectors. In *EACL*.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *ACL*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *WMT*.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *NAACL*.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *ICML*.

Tanja Schultz and Alex Waibel. 1998. Multilingual and crosslingual speech recognition. In *Proc. DARPA Workshop on Broadcast News Transcription and Understanding*. Citeseer.

Lei Shi, Rada Mihalcea, and Mingjun Tian. 2010. Cross language text classification by model translation and semi-supervised learning. In *EMNLP*.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. In *TACL*.

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

Xinyi Wang and Graham Neubig. 2019. Target conditioned sampling: Optimizing data selection for multilingual neural machine translation. In *ACL*.

Xinyi Wang, Hieu Pham, Philip Arthur, and Graham Neubig. 2019a. Multilingual neural machine translation with soft decoupled encoding. In *ICLR*.

Xinyi Wang, Hieu Pham, Paul Mitchel, Antonis Anastasopoulos, Jaime Carbonell, and Graham Neubig. 2019b. Optimizing data usage via differentiable rewards. In *arxiv*.

Yining Wang, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2018. Three strategies to improve one-to-many multilingual translation. In *EMNLP*.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *EMNLP*.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *EMNLP*.

Poorya Zaremoodi, Wray L. Buntine, and Gholamreza Haffari. 2018. Adaptive knowledge sharing in multi-task learning: Improving low-resource neural machine translation. In *ACL*.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low resource neural machine translation. In *EMNLP*.

A Appendix

A.1 Effect of Step-ahead Reward

| Setting | Baseline | MultiDDS | |
		Moving Ave.	Step-ahead
M2O	24.88	25.19	25.26
O2M	16.61	17.17	17.17

Table 5: Ave. BLEU for the Related language group. The step-ahead reward proposed in the paper is better or comparable with the moving average, and both are better than the baseline.

A.2 Hyperparameters

In this section, we list the details of preprocessing and hyperparameters we use for the experiments.

- We use 6 encoder and decoder layers, with 4 attention heads

- The embedding size is set to 512, and the feed-forward layer has a dimension of 1024

- We use the dropout rate of 0.3

- The batch size is set to 9600 tokens

- We use label smoothing with rate of 0.1

- We use the scaled l_2 normalization before residual connection, which is shown to be helpful for small data (Nguyen and Salazar, 2019)

A.3 Dataset statistics

Language	Train	Dev	Test
aze	5.94k	671	903
bel	4.51k	248	664
glg	10.0k	682	1007
slk	61.5k	2271	2445
tur	182k	4045	5029
rus	208k	4814	5483
por	185k	4035	4855
ces	103k	3462	3831

Table 6: Statistics of the related language group.

Language	Train	Dev	Test
bos	5.64k	474	463
mar	9.84k	767	1090
hin	18.79k	854	1243
mkd	25.33k	640	438
ell	134k	3344	4433
bul	174k	4082	5060
fra	192k	4320	4866
kor	205k	4441	5637

Table 7: Statistics of the diverse language group.

A.4 Detailed Results for All Settings

Method	Avg.	aze	bel	glg	slk	tur	rus	por	ces
Uni. ($\tau = \infty$)	22.63	8.81	14.80	25.22	27.32	20.16	20.95	38.69	25.11
Temp. ($\tau = 5$)	24.00	10.42	15.85	27.63	28.38	21.53	21.82	40.18	26.26
Prop. ($\tau = 1$)	24.88	11.20	17.17	27.51	28.85	23.09	22.89	41.60	26.80
MultiDDS	25.26	12.20	18.60	28.83	29.21	22.24	22.50	41.40	27.22
MultiDDS-S	**25.52**	12.20	19.11	29.37	29.35	22.81	22.78	41.55	27.03

Table 8: BLEU score of the baselines and our method on the Related language group for many-to-one translation

Method	Avg.	bos	mar	hin	mkd	ell	bul	fra	kor
Uni. ($\tau = \infty$)	24.81	21.52	9.48	19.99	30.46	33.22	33.70	35.15	15.03
Temp. ($\tau = 5$)	26.01	23.47	10.19	21.26	31.13	34.69	34.94	36.44	16.00
Prop. ($\tau = 1$)	26.68	23.43	10.10	22.01	31.06	35.62	36.41	37.91	16.91
MultiDDS	26.65	25.00	10.79	22.40	31.62	34.80	35.22	37.02	16.36
MultiDDS-S	**27.00**	25.34	10.57	22.93	32.05	35.27	35.77	37.30	16.81

Table 9: BLEU score of the baselines and our method on the Diverse language group for many-to-one translation

Method	Avg.	aze	bel	glg	slk	tur	rus	por	ces
Uni. ($\tau = \infty$)	15.54	5.76	10.51	21.08	17.83	9.94	13.59	30.33	15.35
Temp. ($\tau = 5$)	16.61	6.66	11.29	21.81	18.60	11.27	14.92	32.10	16.26
Prop. ($\tau = 1$)	15.49	4.42	5.99	14.92	17.37	12.86	16.98	34.90	16.53
MultiDDS	17.17	6.24	11.75	21.46	20.67	11.51	15.42	33.41	16.94
MultiDDS-S	**17.32**	6.59	12.39	21.65	20.61	11.58	15.26	33.52	16.98

Table 10: BLEU score of the baselines and our method on the Related language group for one-to-many translation

Method	Avg.	bos	mar	hin	mkd	ell	bul	fra	kor
Uni. ($\tau = \infty$)	16.86	14.12	4.69	14.52	20.10	22.87	25.02	27.64	5.95
Temp. ($\tau = 5$)	17.94	14.73	4.93	15.49	20.59	24.82	26.60	29.74	6.62
Prop. ($\tau = 1$)	16.79	6.93	3.69	10.70	15.77	26.69	29.59	33.51	7.49
MultiDDS	**18.40**	14.91	4.83	14.96	22.25	24.80	27.99	30.77	6.75
MultiDDS-S	18.24	14.02	4.76	15.68	21.44	25.69	27.78	29.60	7.01

Table 11: BLEU score of the baselines and our method on the Diverse language group for one-to-many translation