Adaptively Scheduled Multitask Learning: The Case of Low-Resource Neural Machine Translation

Poorya Zaremoodi[1,2] **Gholamreza Haffari**[1]
[1]Monash University, Melbourne, Australia
[2]CSIRO Data61, Sydney, Australia
first.last@monash.edu

Abstract

Neural Machine Translation (NMT), a data-hungry technology, suffers from the lack of bilingual data in low-resource scenarios. Multitask learning (MTL) can alleviate this issue by injecting inductive biases into NMT, using auxiliary syntactic and semantic tasks. However, an effective *training schedule* is required to balance the importance of tasks to get the best use of the training signal. The role of training schedule becomes even more crucial in *biased-MTL* where the goal is to improve one (or a subset) of tasks the most, e.g. translation quality. Current approaches for biased-MTL are based on brittle *hand-engineered* heuristics that require trial and error, and should be (re-)designed for each learning scenario. To the best of our knowledge, ours is the first work on *adaptively* and *dynamically* changing the training schedule in biased-MTL. We propose a rigorous approach for automatically reweighing the training data of the main and auxiliary tasks throughout the training process based on their contributions to the generalisability of the main NMT task. Our experiments on translating from English to Vietnamese/Turkish/Spanish show improvements of up to +1.2 BLEU points, compared to strong baselines. Additionally, our analyses shed light on the dynamic of needs throughout the training of NMT: from syntax to semantic.

1 Introduction

While Neural Machine Translation (NMT) is known for its ability to learn end-to-end without any need for many brittle design choices and hand-engineered features, it is notorious for its demand for large amounts of bilingual data to achieve reasonable translation quality (Koehn and Knowles, 2017). Recent work has investigated multitask learning (MTL) for injecting inductive biases from auxiliary syntactic and/or semantic tasks into NMT to improve its generalisation (Zaremoodi and Haffari, 2018; Zaremoodi et al., 2018; Kiperwasser and Ballesteros, 2018).

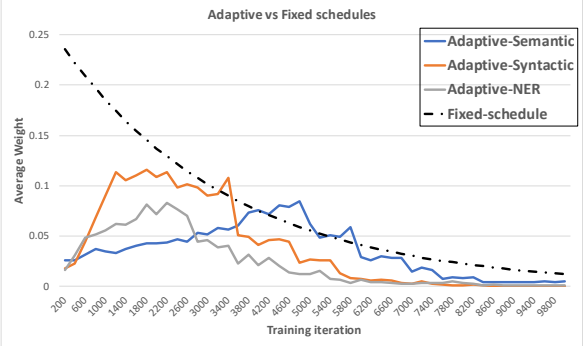


Figure 1: The dynamic in the relative importance of named entity recognition, syntactic parsing, and semantic parsing as the auxiliary tasks for the main machine translation task (based on our experiments in §3). The plot shows our proposed adaptive scheduling vs fixed scheduling (Kiperwasser and Ballesteros, 2018) (scaled down for better illustration).

The majority of the MTL literature has focused on investigating how to share common knowledge among the tasks through tying their parameters and joint training using standard algorithms. However, a big challenge of MTL is how to get the best signal from the tasks by changing their importance in the training process aka *training schedule*; see Figure 1.

Crucially, a proper training schedule would encourage positive transfer and prevent negative transfer, as the inductive biases of the auxiliary tasks may interfere with those of the main task leading to degradation of generalisation capabilities. Most of the works on training schedule focus on *general* MTL where the goal is to improve the performance of *all* tasks. They are based on addressing the imbalance in task difficulties and co-evolve easy and difficult tasks uniformly (performance-wise). These methods

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 177–186
Hong Kong, China, November 4, 2019.
www.aclweb.org/anthology/D19-56%2d

achieve competitive performance with existing single-task models of each task, and not necessarily much better performance (Chen et al., 2018; Guo et al., 2018b). On the other hand, *biased-MTL* focuses on the *main* task to achieve higher improvements on it. (Zaremoodi and Haffari, 2018) has proposed a fixed training schedule to balance out the importance of the main NMT task vs auxiliary task to improve NMT the most. (Kiperwasser and Ballesteros, 2018) has shown the effectiveness of a changing training schedule through the MTL process. However, their approach is based on *hand-engineered* heuristics, and should be (re-)designed and fine-tuned for every change in tasks or even training data.

In this paper, for the first time to the best of our knowledge, we propose a method to *adaptively* and *dynamically* set the importance weights of tasks for *biased-MTL*. By using *influence functions* from robust statistics (Cook and Weisberg, 1980; Koh and Liang, 2017), we adaptively examine the influence of training instances inside mini-batches of the tasks on the generalisation capabilities on the main task. The generalisation is measured as the performance of the main task on a validation set, separated from the training set, in each parameter update step *dynamically*. In this paper, we consider translation as the main task along with syntactic and semantic auxiliary tasks, and re-weight instances in such a way to maximise the performance of the translation task. As our method is general and does not rely on hand-engineered heuristics, it can be used for effective learning of multitask architectures beyond NMT.

We evaluate our method on translating from English to Vietnamese/Turkish/Spanish, with auxiliary tasks including syntactic parsing, semantic parsing, and named entity recognition. Compared to strong training schedule baselines, our method achieves considerable improvements in terms of BLEU score. Additionally, our analyses on the weights assigned by the proposed training schedule show that although the dynamic of weights are different for different language pairs, the underlying pattern is gradually altering tasks importance from syntactic to semantic-related tasks.

In summary, our main contributions to MTL and low-resource NMT are as follows:

- We propose an effective training schedule for biased-MTL that adaptively and dynamically set the importance of tasks throughout the training to improve the main task the most.
- We extensively evaluate on three language pairs, and experimental results show that our model outperforms the hand-engineered heuristics.
- We present an analysis to better understand and shed light on the dynamic of needs of an NMT model during training: from syntax to semantic.

2 Learning to Reweigh Mini-Batches

Suppose we are given a set of a main task along with $K-1$ auxiliary tasks, each of which with its own training set $\{(\boldsymbol{x}_i^{(k)}, \boldsymbol{y}_i^{(k)})\}_{i=1}^{N_k}$. In multitask formulation, parameters are learned by maximising the log-likelihood objective:

$$\arg\max_{\boldsymbol{\Theta}_{mtl}} \sum_{k=1}^{K} \sum_{i=1}^{N_k} w_i^{(k)} \log P_{\boldsymbol{\Theta}_{mtl}}(\boldsymbol{y}_i^{(k)} | \boldsymbol{x}_i^{(k)}).$$

Without loss of generality, let us assume we use minibatch-based stochastic gradient descent (SGD) to train the parameters of the multitask architecture. In standard multitask learning $w_i^{(k)}$ is set to 1, assuming all of the tasks and their training instances have the same importance. Conceptually, these weights provide a mechanism to control the influence of the data instances from auxiliary tasks in order to maximise the benefit in the generalisation capabilities of the main task. Recently, (Zaremoodi and Haffari, 2018; Kiperwasser and Ballesteros, 2018) have proposed *hand-engineered* heuristics to set the importance weights and change them dynamically throughout the training process, e.g., iterations of the stochastic gradient descent (SGD). However, there is no guarantee that these fixed schedules give rise to learning the best inductive biases from the auxiliary tasks for the main task.

Our main idea is to determine the importance weights $w_i^{(k)}$ for *each* training instance based on its contribution to the generalisation capabilities of the MTL architecture for machine translation, measured on a validation set D^{val} separated from the training set. As shown in Figure 2, at each parameter update iteration for the MTL architecture, the MTL training mini-batch is the concatenation of single mini-batches from all MTL tasks. We then assign an Adaptive Importance Weight

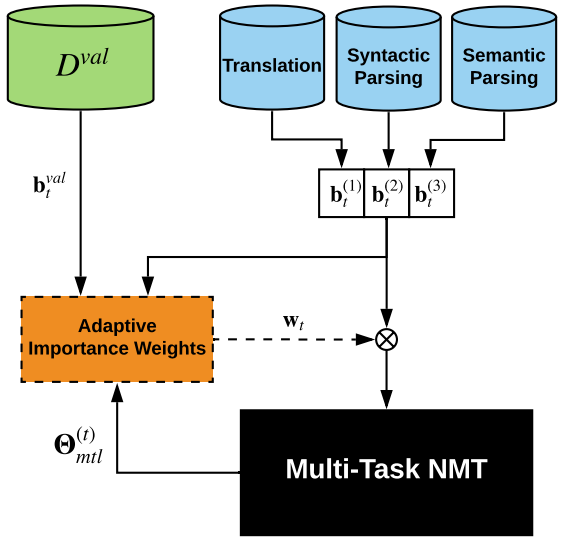


Figure 2: High-level idea for training an MTL architecture using adaptive importance weights (AIWs). Here, translation is the main task along with syntactic and semantic parsing as auxiliary linguistic tasks.

(AIW) to *each* training instance in the MTL mini-batch, regardless from the task which they come from. In the experiments of §3, we will see that our proposed method *automatically* finds interesting patterns in how to best make use of the data from the auxiliary and main tasks, e.g. it starts by assigning higher weights (on average) to syntactic parsing which is then shifted to semantic parsing.

More specifically, we learn the AIWs based on the following optimisation problem:

$$\arg\min_{\hat{\boldsymbol{w}}} \; -\sum_{(\boldsymbol{x},\boldsymbol{y})\in D^{val}} \log P_{\hat{\mathbf{\Theta}}_{mtl}(\hat{\boldsymbol{w}})}(\boldsymbol{x}|\boldsymbol{y}) \quad (1)$$

$$\hat{\mathbf{\Theta}}_{mtl}(\hat{\boldsymbol{w}}) := \mathbf{\Theta}_{mtl}^{(t)} + \eta \sum_{k=1}^{K} \sum_{i=1}^{|\boldsymbol{b}^{(k)}|} \hat{w}_i^{(k)} \nabla \log P_{\mathbf{\Theta}_{mtl}^{(t)}}(\boldsymbol{y}_i^{(k)}|\boldsymbol{x}_i^{(k)}) \quad (2)$$

where $\hat{w}_i^{(k)}$ is the *raw* weight of the ith training instance in the mini-batch $\boldsymbol{b}^{(k)}$ of the kth task, $\hat{\mathbf{\Theta}}_{mtl}$ is the resulting parameter in case SGD update rule is applied on the current parameters $\mathbf{\Theta}_{mtl}^{(t)}$ using instances weighted by $\hat{\boldsymbol{w}}$. Following (Ren et al., 2018), we zero out negative raw weights, and then normalise them with respect to the other instances in the MTL training mini-batch to obtain the AIWs: $w_i^{(k)} = \frac{\tilde{w}_i^{(k)}}{\sum_{k'}\sum_{i'}\tilde{w}_{i'}^{(k')}}$ where $\tilde{w}_i^{(k)} = \text{ReLU}(\hat{w}_i^{(k)})$.

In the preliminary experiments, we observed that using $w_i^{(k)}$ as AIW does not perform well. We speculate that a small validation set does not provide a good estimation of the generalisation, hence influence of the training instances. This is exacerbated as we approximate the validation set by only one of its mini-batches for the computational efficiency. Therefore, we hypothesise that the computed weights should not be regarded as the final verdict for the usefulness of the training instances. Instead, we regarded them as *rewards* for enhancing the training signals of instances that lead to a lower loss on the validation set. Hence, we use $1 + w_i^{(k)}$ as our AIWs in the experiments. The full algorithm is in Algorithm 1.

Algorithm 1 Adaptively Scheduled Multitask Learning

while t=0 ... T-1 **do**
  $\boldsymbol{b}^{(1)}, .., \boldsymbol{b}^{(K)} \leftarrow \text{SampleMB}(\boldsymbol{D}^{(1)}, .., \boldsymbol{D}^{(K)})$
  $\boldsymbol{b}^{(val)} \leftarrow \text{SampleMB}(\boldsymbol{D}^{(val)})$
  ▷ Step 1: Update model with initialised weights
  $\ell_i^{(k)} \leftarrow -\log P_{\mathbf{\Theta}_{mtl}^t}(\boldsymbol{y}_i^{(k)}|\boldsymbol{x}_i^{(k)})$ ▷ Forward
  $\hat{w}_{i,0}^{(k)} \leftarrow 0$ ▷ Initialise weights
  $\mathcal{L}_{trn} \leftarrow \sum_{k=1}^{K}\sum_{i=1}^{|\boldsymbol{b}^{(k)}|} \hat{w}_{i,0}^{(k)} \ell_i^{(k)}$
  $g_{trn} \leftarrow \text{Backward}(\mathcal{L}_{trn}, \mathbf{\Theta}_{mtl}^t)$
  $\hat{\mathbf{\Theta}}_{mtl}^t = \mathbf{\Theta}_{mtl}^t + \eta g_{trn}$
  ▷ Step 2: Calculate loss of the updated model on validation MB
  $\mathcal{L}_{val} = -\sum_{i=1}^{|\boldsymbol{b}^{val}|} \log P_{\hat{\mathbf{\Theta}}_{mtl}^t}(\boldsymbol{y}_i|\boldsymbol{x}_i)$
  ▷ Step 3: Calculate raw weights.
  $g_{val} \leftarrow \text{Backward}(\mathcal{L}_{val}, \hat{w}_0^{(k)})$
  $\hat{w}^{(k)} = g_{val}$
  ▷ Step 4: Normalise weights to get AIWs
  $\tilde{w}_i^{(k)} = \text{ReLU}(\hat{w}_i^{(k)})$
  $w_i^{(k)} = \frac{\tilde{w}_i^{(k)}}{\sum_{k'}\sum_{i'}\tilde{w}_{i'}^{(k')}} + 1$
  ▷ Step 5: Update MTL with AIWs
  $\hat{\mathcal{L}}_{trn} \leftarrow \sum_{k=1}^{K}\sum_{i=1}^{|\boldsymbol{b}^{(k)}|} w_i^{(k)} \ell_i^{(k)}$
  $\hat{g}_{trn} \leftarrow \text{Backward}(\hat{\mathcal{L}}_{trn}, \mathbf{\Theta}_{mtl}^t)$
  $\mathbf{\Theta}_{mtl}^{t+1} = \mathbf{\Theta}_{mtl}^t + \eta \hat{g}_{trn}$
end while

Implementation Details. As exactly solving the optimisation problem in Eq. (1) is challenging, we resort to an approximation and consider the raw weights as the gradient of the validation loss

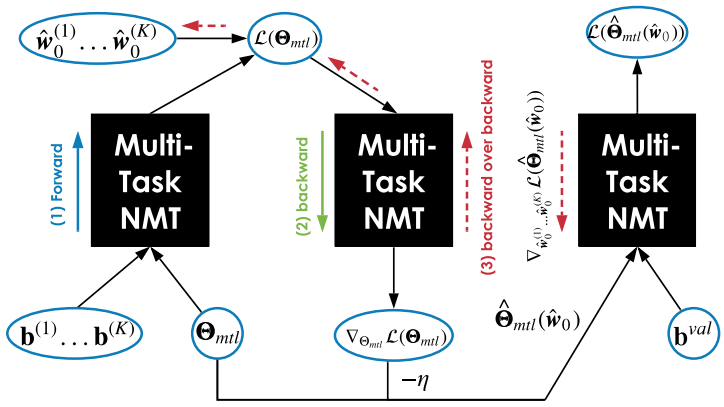


Figure 3: Computation graph of the proposed method for adaptively determining weights.

wrt the training instances' weights around zero. This is a notion called *influence* in robust statistics (Cook and Weisberg, 1980; Koh and Liang, 2017).

More concretely, let us define the loss function $\mathcal{L}(\hat{\Theta}_{mtl}) := -\sum_{i=1}^{|b^{val}|} \log P_{\hat{\Theta}_{mtl}}(y_i|x_i)$, where b^{val} is a minibatch from the validation set. The training instances' raw weights, i.e. influences, are then calculated using the chain rule:

$$\hat{w} = \nabla_{\hat{w}_0} \mathcal{L}(\hat{\Theta}_{mtl}(\hat{w}_0))\Big|_{\hat{w}_0=\mathbf{0}}$$
$$= \nabla_{\hat{\Theta}_{mtl}} \mathcal{L}(\hat{\Theta}_{mtl})\Big|_{\hat{\Theta}_{mtl}=\Theta_{mtl}^{(t)}} \cdot \nabla_{\hat{w}_0} \hat{\Theta}_{mtl}(\hat{w}_0)\Big|_{\hat{w}_0=\mathbf{0}}$$

The last term $\nabla_{\hat{w}_0} \hat{\Theta}_{mtl}$ involves backpropagation through $\hat{\Theta}_{mtl}$ wrt $\hat{w}_0$, which according to Eq. (2), involves an inner backpropagation wrt Θ_{mtl}. The computation graph is depicted in Figure 3.

3 Experiments

3.1 Bilingual Corpora

We use three language-pairs, translating from English to Vietnamese (Vi), Turkish (Tr) and Spanish (Es). We have chosen them to analyse the effect of adaptive mini-batch weighting on languages with different underlying linguistic structures. The structure of Vietnamese and Spanish is generally subject-verb-object (SVO) while Turkish follows subject-object-verb (SOV) structure. Although Spanish is not a low-resource language we have chosen it because of available accurate POS taggers and Named-Entity recognisers required for some of the analyses. For each pair, we use BPE (Sennrich et al., 2016) with 40K types on the union of the source and target vocabularies. We use the Moses toolkit (Koehn et al., 2007) to filter out pairs where the number of tokens is more than 250 and pairs with a source/target length ratio higher than 1.5. For fair comparison, we add the Val data used in the AIW-based approach to the training set of the competing baselines.

- English-Vietnamese: we use the preprocessed version of IWSLT 2015 corpus (Cettolo et al., 2015) provided by (Luong and Manning, 2015). It consists of about 133K training pairs from the subtitles of TED and TEDx talks and their translations. We use "tst2013" as the test set and "tst2012" is divided and used as validation and meta-validation sets (with the ratio 2 to 1).

- English-Turkish: we use WMT parallel corpus (Bojar et al., 2016) with about 200K training pairs gathered from news articles. "newstest2016", "newstest2017" and "newstest2018" parts are used as validation, meta-validation and test set.

- English-Spanish: we have used the first 150K training pairs of Europarl corpus (Koehn, 2005). "newstest2011", "newstest2012" and "newstest2013" parts are used as validation, meta-validation and test set, respectively.

3.2 Auxiliary tasks

Following (Zaremoodi and Haffari, 2018), we have chosen following auxiliary tasks to inject the syntactic and semantic knowledge to improve NMT:

- Named-Entity Recognition (NER): we use CONLL shared task[1] data. This dataset is

[1] https://www.clips.uantwerpen.be/conll2003/ner

	En→Vi		En→Tr		En→Es			
	BLEU		BLEU		BLEU		METEOR	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test
MT only	22.83	24.15	8.55	8.5	14.49	13.44	31.3	31.1
MTL with Fixed Schedule								
+ Uniform	23.10	24.81	9.14	8.94	12.81	12.12	29.6	29.5
+ Biased (Constant)[†‡]	23.42	25.22	10.06	9.53	15.14	14.11	31.8	31.3
+ Exponential[‡]	23.45	25.65	9.62	9.12	12.25	11.62	28.0	28.1
+ Sigmoid[‡]	23.35	25.36	9.55	9.01	11.55	11.34	26.6	26.9
MTL with Adaptive Schedule								
+ Biased + **AIW**	23.95	25.75	10.67	10.25	11.23	10.66	27.5	27.4
+ Uniform + **AIW**	24.38	**26.68**	11.03	**10.81**	16.05	**14.95**	33.0	**32.5**

Table 1: Results for three language pairs. "+ AIW" indicates Adaptive Importance Weighting is used in training. [†]: Proposed in (Zaremoodi and Haffari, 2018), [‡]: Proposed in (Kiperwasser and Ballesteros, 2018).

consists of a collection of newswire articles from the Reuters Corpus.

- Syntactic Parsing: we use Penn TreeBank parsing with the standard split (Marcheggiani and Titov, 2017). This task is casted to SEQ2SEQ transduction by linearising constituency trees (Vinyals et al., 2015)

- Semantic Parsing: we use Abstract Meaning Representation (AMR) corpus Release 2.0[2] linearised by the method proposed in (Konstas et al., 2017). This corpus is gathered from from newswire, weblogs, web discussion forums and broadcast conversations.

3.3 MTL architecture and training schedule

Since partial-sharing has been shown to be more effective than full sharing (Liu et al., 2017; Guo et al., 2018a; Zaremoodi and Haffari, 2018), we use the MTL architecture proposed in (Zaremoodi and Haffari, 2018). We use three stacked LSTM layers in encoders and decoders. For En→Vi and En→Tr, one/two layer(s) are shared among encoders/decoders while for En→Es, two/one layer(s) are shared among encoders/decoders. The LSTM dimensions, batch size and dropout are set to 512, 32 and 0.3, respectively. We use Adam optimiser (Kingma and Ba, 2014) with the learning rate of 0.001. We train models for 25 epochs and save the best model based on the perplexity on the validation (Val) set. We have implemented the methods using PyTorch on top of OpenNMT (Klein et al., 2017).

[2]https://catalog.ldc.upenn.edu/LDC2017T10

Fixed *hand-engineered* schedule baselines. We use different MTL scheduling strategies where at each update iteration:

- **Uniform**: Selects a random mini-batch from all of the tasks;

- **Biased** (Zaremoodi and Haffari, 2018): Selects a random mini-batch from the translation task (bias towards the main task) and another one for a randomly selected task.

We also use schedules proposed in (Kiperwasser and Ballesteros, 2018). They consider a slope parameter[3] α and the fraction of training epochs done so far $t = sents/||corpus||$. The schedules determine the probability of selecting each of the tasks as the source of the next training pair. In each of these schedules the probability of selecting the *main task* is:

- **Constant:** $P_m(t) = \alpha$; When α is set to 0.5, it is similar to the Biased schedule we have seen before.

- **Exponential:** $P_m(t) = 1 - e^{-\alpha t}$; In this schedule the probability of selecting the main task increases exponentially throughout the training.

- **Sigmoid:** $P_m(t) = \frac{1}{1+e^{-\alpha t}}$; Similar to the previous schedule, the probability of selecting the main task increases, following a sigmoid function.

In each of these schedules, the rest of the probability is uniformly divided among the *remaining*

[3]Following their experiments, we set α to 0.5.

tasks. By using them, a mini-batch can have training pairs from different tasks which makes it inefficient for partially shared MTL models. Hence, we modified these schedules to select the source of the next training mini-batch.

Combination of Adaptive and Fixed schedules As mentioned in Section 2, we assign an AIW to each training instance inside mini-batches of *all* tasks, i.e. applying AIWs on top of Uniform schedule. Additionally, we also apply it on top of Biased schedule to analyse the effect of the combination of AIWs (for instances) and a *hand-engineered* heuristic (for mini-batch selection).

3.4 Results and Analysis

Table 1 reports the results for baselines and the proposed method[4]. As seen, our method has made better use of the auxiliary tasks and achieved the highest performance (see Section 3.4 for an analysis of the generated translations). It shows that while some of the *heuristic*-based schedules are beneficial, our proposed Adaptive Importance Weighting approach outperforms them. There reasons are likely that the *hand-engineered* strategies do not consider the state of the model, and they do not distinguish among the auxiliary tasks.

It is interesting to see that the Biased schedule is beneficial for standard MTL, while it is harmful when combined with the AIWs. The standard MTL is not able to select training signals on-demand, and using a biased heuristic strategy improves it. However, our weighting method can selectively filter out training signals; hence, it is better to provide all of the training signals and leave the selection to the AIWs.

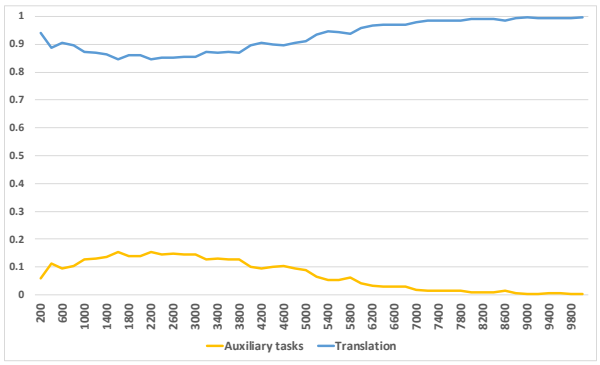
Analysis on how/when auxiliary tasks have been used? This analysis aims to shed light on how AIWs control the contribution of each task through the training. As seen, our method has the best result when it is combined with the Uniform MTL schedule. In this schedule, at each update iteration, we have one mini-batch from each of the tasks, and AIWs are determined for all of the training pairs in these mini-batches. For this analysis, we divided the training into 200 update iteration chunks. In each chunk, we compute the average weights assigned to the training pairs of each task.

[4]METEOR score (Denkowski and Lavie, 2014) is reported only for Spanish as it is the only target languages in our experiments which is supported by it.

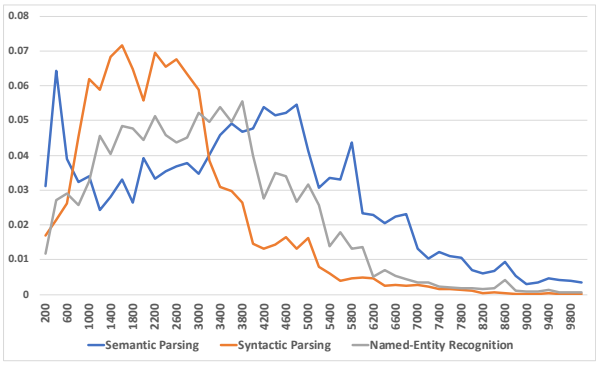
Figure 1 shows the results of this analysis for the MTL model trained with En→Vi as the main task. and Figure 4 shows the results of this analysis for En→Es and En→Tr. Also, it can be seen that at the beginning of the training the Adaptive Importance Weighting mechanism gradually increases the training signals which come from the auxiliary tasks. However, after reaching a certain point in the training, it will gradually reduce the auxiliary training signals to concentrate more on the adaptation to the main task. It can be seen that the weighting mechanism distinguishes the importance of auxiliary tasks. More interestingly, it can be seen that for the English→Turkish, the contribution of NER task is more than the syntactic parsing while for the other languages we have seen the reverse. It shows that our method can *adaptively* determine the contribution of the tasks by considering the demand of the main translation task.

As seen, it gives more weight to the syntactic tasks at the beginning of the training while it gradually reduces their contribution and increases the involvement of the semantics-related task. We speculate the reason is that at the beginning of the training, the model requires more lower-level linguistic knowledge (e.g. syntactic parsing and NER) while over time, the needs of model gradually change to higher-level linguistic knowledge (e.g. semantic parsing).

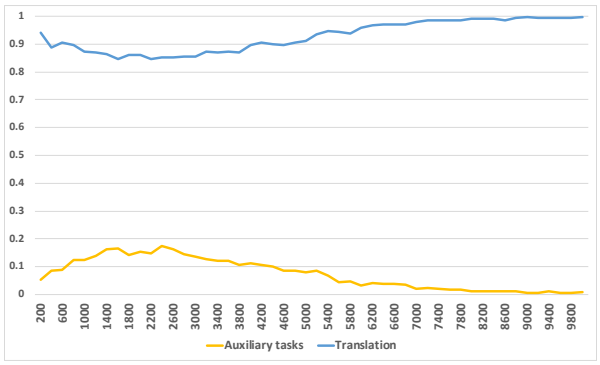
Analysis of The Effect of Auxiliary Tasks on The Generated Translations In this analysis, we want to take a deeper look at the generated translations and see how the proposed method improved the quality of the translations. More specifically, we want to compare the number of words in the gold translations which are missed in the generated translations produced by the following systems: (i) MT only; (ii) MTL-Biased; (iii) MTL-Uniform + AIW. To find out what kind of knowledge is missed in the process of generating the translations, we categorised words by their Part-of-Speech tags and named-entities types. We have done this analysis on En→Es language pair as there are accurate annotators for the Spanish language. We use Stanford POS tagger (Toutanova et al., 2003) and named-entity recogniser (Finkel et al., 2005) to annotate Spanish gold translations. Then, we categorised the missed words in the generated translations concerning these tags, and count the number of missed words in each category. Figure 5 depicts the result. As seen in



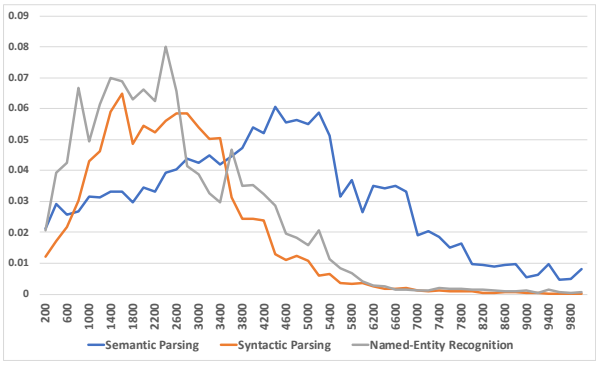
(a) Translation task (En→Es) vs. auxiliary tasks.



(b) Auxiliary tasks vs. each other.



(c) Translation task (En→Tr) vs. auxiliary tasks.



(d) Auxiliary tasks vs. each other.

Figure 4: Weights assigned to the training pairs of different tasks (averaged over 200 update iteration chunks). Y-axis shows the average weight and X-axis shows the number of update iteration. In the top figures, the main translation task is English→Spanish while in the bottom ones it is English→Turkish.

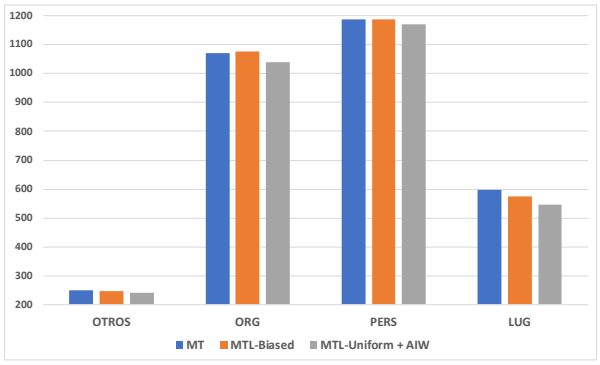
Figure 5a, the knowledge learned from auxiliary tasks helps the MTL model to miss less number of named-entities during translation. Moreover, AIWs help the MTL model further by making better use of the knowledge conveyed in the auxiliary tasks. We can see the same pattern for the POS of missed words. As seen, for most POS categories, the standard MTL has missed less number of words in comparison with the MT only baseline. Furthermore, our method helps the MTL model to miss even less amount of words in every of the POS categories (specifically in Noun and Preposition categories). We speculate the reason is that the AIWs makes it possible to control the contribution of each of the auxiliary tasks separately and taking into account the demand of the model at each stage of the training procedure.

4 Related Work

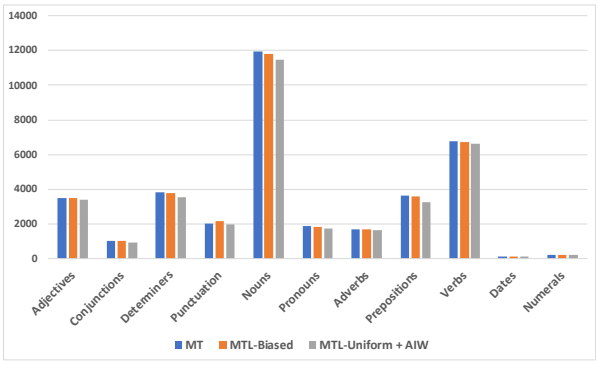
Multitask learning (Caruana, 1997) has been used for various NLP problems, e.g. machine translation (Dong et al., 2015), dependency parsing (Peng et al., 2017), key-phrase boundary classification (Augenstein and Søgaard, 2017), video captioning (Pasunuru and Bansal, 2017), Chinese word segmentation, and text classification problem (Liu et al., 2017). For the case of low-resource NMT, (Niehues and Cho, 2017) has explored the use of part-of-speech and named-entity recognition in improving NMT. (Kiperwasser and Ballesteros, 2018) has investigated part-of-speech tagging and dependency parsing tasks, and (Zaremoodi et al., 2018; Zaremoodi and Haffari, 2018) have tried syntactic parsing, semantic parsing, and named-entity recognition tasks.

The current research on MTL is focused on encouraging positive transfer and preventing the negative transfer phenomena in two lines of research: (1) Architecture design: works in this area try to learn effective parameter sharing among tasks (Ruder et al., 2017; Zaremoodi et al., 2018); (2) Training schedule: works in this area, including ours, focus on setting the importance of tasks.

Training schedule is the beating heart of MTL, and has a critical role in the *performance* of the resulted model. Since there are more than one task involved in MTL, the performance is measured differently in different MTL flavours: (1)



(a) Named-Entities



(b) Part-of-Speech tags

Figure 5: The number of words in the gold English→Spanish translation which are missed in the generated translations (lower is better). Missed words are categorised by their tags (Part-of-Speech and named-entity types).

general-MTL aims to improve performance of *all* tasks; (2) *biased-MTL* aims to improve *one* (or a subset) of tasks the most. Training schedules designed for the global-MTL are focused on co-evolving easy and difficult tasks uniformly. These methods are designed to achieve competitive performance with existing single-task models of each task (Chen et al., 2018; Guo et al., 2018b). On the other hand, training schedules for biased-MTL focus on achieving higher improvements on the main task, and our method belongs to this category.

Training schedules can be fixed/dynamic throughout the training and be hand-engineered/adaptive. (Zaremoodi and Haffari, 2018) has made use of a fixed hand-engineered schedule for improving low-resource NMT with auxiliary linguistic tasks. Recently, (Guo et al., 2019) has proposed an adaptive way to compute the importance weights of tasks. Instead of manual tuning of importance weights via a large grid search, they model the performance of each set of weights as a sample from a Gaussian Process (GP), and search for optimal values. In fact, their method is not completely adaptive as a strong prior needs to be set for the main task. This method can be seen as a guided yet computationally exhaustive trial-and-error where in each trial, MTL models need to be re-trained (from scratch) with the sampled weights. Moreover, the weight of tasks are fixed throughout the training. At least, for the case of low-resource NMT, it has been shown that *dynamically* changing the weights throughout the training is essential to make better use of auxiliary tasks (Kiperwasser and Ballesteros, 2018). (Kiperwasser and Ballesteros, 2018) has proposed *hand-engineered* training schedules for MTL in NMT, where they dynamically change the importance of the main task vs the auxiliary tasks throughout the training process. While their method relies on *hand-engineered* schedules which should be tuned by trial-and-error, our proposed method *adaptively* and *dynamically* sets the importance of the tasks and learn the MTL model in the course of a single training run.

5 Conclusions

This paper presents a rigorous approach for adaptively and dynamically changing the training schedule in biased-MTL to make the best use of auxiliary tasks. To balance the importance of the auxiliary tasks vs. the main task, we re-weight training data of tasks to adjust their contributions to the generalisation capabilities of the resulted model on the main task. In this paper, we consider low-resource translation as the main task along with syntactic and semantic auxiliary tasks. Our experimental results on English to Vietnamese/Turkish/Spanish show up to +1.2 BLEU score improvement compared to strong baselines. Additionally, the analyses show that the proposed method *automatically* finds a schedule which puts more importance to the auxiliary syntactic tasks at the beginning while gradually it alters the importance toward the auxiliary semantic task. As this method does not rely on hand-engineered heuristics, as a future work, we want to apply it for effective learning of multitask architectures beyond NMT.

Acknowledgement

This work is supported by CSIRO Data61 through a PhD Fellowship to P. Z., and by an Amazon Re-

search Award to G. H. This work is partly sponsored by DARPA through the contract no FA8750-19-2-0501. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the US government. This work was supported by the Multi-modal Australian ScienceS Imaging and Visualisation Environment (MASSIVE) (www.massive.org.au) through computational infrastructure. We would like to thank anonymous reviewers for their insightful comments.

References

Isabelle Augenstein and Anders Søgaard. 2017. Multi-task learning of keyphrase boundary classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 341–346.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *ACL 2016 First Conference On Machine Translation (WMT16)*, pages 131–198. The Association for Computational Linguistics.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2015. The iwslt 2015 evaluation campaign. In *IWSLT 2015, International Workshop on Spoken Language Translation*.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pages 793–802.

R Dennis Cook and Sanford Weisberg. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2018a. Soft layer-specific multi-task summarization with entailment and question generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 687–697.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2019. Autosem: Automatic task selection and mixing in multi-task learning. *CoRR*, abs/1904.04153.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. 2018b. Dynamic task prioritization for multitask learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 270–287.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 177–180.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39.

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*, pages 1885–1894. JMLR. org.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 146–157.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1–10.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*, Da Nang, Vietnam.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. In *Proceedings of the Second Conference on Machine Translation*, pages 80–89.

Ramakanth Pasunuru and Mohit Bansal. 2017. Multi-task video captioning with video and entailment generation. In *Proceedings of ACL*.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2048.

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 4331–4340.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice networks: Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 conference of the North American chapter of the association for computational linguistics on human language technology-volume 1*, pages 173–180.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, pages 2773–2781.

Poorya Zaremoodi, Wray L. Buntine, and Gholamreza Haffari. 2018. Adaptive knowledge sharing in multi-task learning: Improving low-resource neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics,*, pages 656–661.

Poorya Zaremoodi and Gholamreza Haffari. 2018. Neural machine translation for bilingually scarce scenarios: a deep multi-task learning approach. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1356–1365.