Model-Level Dual Learning

Yingce Xia [1 2] **Xu Tan** [2] **Fei Tian** [2] **Tao Qin** [2] **Nenghai Yu** [1] **Tie-Yan Liu** [2]

Abstract

Many artificial intelligence tasks appear in dual forms like English↔French translation and speech↔text transformation. Existing dual learning schemes, which are proposed to solve a pair of such dual tasks, explore how to leverage such dualities from data level. In this work, we propose a new learning framework, model-level dual learning, which takes duality of tasks into consideration while designing the architectures for the primal/dual models, and ties the model parameters that playing similar roles in the two tasks. We study both symmetric and asymmetric model-level dual learning. Our algorithms achieve significant improvements on neural machine translation and sentiment analysis.

1. Introduction

Joint learning of multiple tasks has attracted much attention in machine learning community, and several learning paradigms have been studied to explore the task correlations from different perspective. Multi-task learning (Luong et al., 2016; Zhang & Yang, 2017) is a paradigm that learns a problem together with other related problems at the same time, using a shared representation. This often leads to a better model for the main task, because it allows the learner to use the commonality among the tasks. Transfer learning (Pan & Yang, 2010) focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. Recently, a new paradigm, dual learning (He et al., 2016a; Xia et al., 2017b; Yi et al., 2017; Lin et al., 2018), is proposed to leverage the symmetric structure of some learning problems, such as English↔French translation, image↔text transformation, and speech↔text transformation, and achieves promising results in various AI tasks (He et al., 2016a; Yi et al., 2017; Tang et al., 2017).

[1]School of Information Science and Technology, University of Science and Technology of China, Hefei, China [2]Microsoft Research, Beijing, China. Correspondence to: Tao Qin <taoqin@microsoft.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

Dual learning algorithms have been proposed for different learning settings. (He et al., 2016a; Yi et al., 2017) focus on the unsupervised setting and learn from unlabeled data: given an unlabeled sample $x \in \mathcal{X}$, the primal model $f : \mathcal{X} \mapsto \mathcal{Y}$ first maps it to a sample $y \in \mathcal{Y}$, the dual model $g : \mathcal{Y} \mapsto \mathcal{X}$ maps y to a new sample $\hat{x} \in \mathcal{X}$, and then the distortion between x and $\hat{x}$ is used as the feedback signal to optimize f and g. (Xia et al., 2017b) focuses on the supervised setting and conducts joint learning from labeled data by adding an additional probabilistic constraint $P(x)P(y|x;f) = P(y)P(x|y;g)$ on any (x, y) data pair implied by structure duality. This probabilistic constraint is also utilized in inference process (Xia et al., 2017a). Inspired by the law of total probability, (Wang et al., 2018) studies dual transfer learning, in which one model in a pair of dual tasks is used to enhance the training of the other model. Although those algorithms consider different settings, they all consider duality at data level, characterized by either the reconstruction error of unlabeled samples (He et al., 2016a; Yi et al., 2017), the joint probability of data pairs (Xia et al., 2017b; Wang et al., 2018), or the marginal probability of data samples (Wang et al., 2018).

We find that many tasks are of structural duality/symmetry not only in data level, but also in model level. Take neural machine translation (briefly, NMT) (Xia et al., 2017c;d), which attacks the problem of translating a source-language sentence $x \in \mathcal{X}$ to a target-language sentence $y \in \mathcal{Y}$, as an example here. Such kinds of tasks are usually handled by an encoder-decoder framework. We summarize an one-layer LSTM (Hochreiter & Schmidhuber, 1997) based model in Figure 1 and the other model structures like CNN (Gehring et al., 2017), self-attention (Vaswani et al., 2017) can be similarly formulated. The encoder $\varphi^c_{p,X}$ takes a source sentence x as an input and outputs a set of hidden representations $h^X_i \in \mathcal{H}\ \forall i \in [T_x]$, where $\mathcal{H}$ is the space of hidden representations, and T_x is the length of sentence x. Next, the decoder $\varphi_{p,Y}$ computes the hidden state h^Y_j by taking the previous h^Y_{j-1}, y_{j-1} and the $h^X_i\ \forall i \in [T_x]$ from the encoder as inputs. $\varphi_{p,Y}$ consists of three parts: $\varphi^z_{p,Y}$ is the attention model used to generate Z^X_j (e.g., the contextual information); $\varphi^c_{p,Y}$ is used to compute h^Y_j; and $\varphi^h_{p,Y}$ is used to map h^Y_j to space $\mathcal{Y}$, which is usually a softmax operator.

The aforementioned processes can be mathematically for-

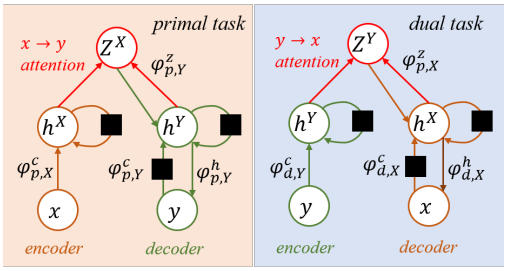


Figure 1. An architecture of existing encoder-decoder models. The black square indicates an optional delay of a single time step. Subscripts "p" and "d" indicate primal and dual models respectively. Subscripts "X" and "Y" denote the two languages. The unfold version can be found at Appendix A. All appendices are left in the supplementary document due to space limitation.

mulated as follows: $\forall i \in [T_x]$ and $\forall j \in [T_y]$ (T_y is the length of y),

$$h_i^X = \varphi_{p,X}^c(x_i, h_{i-1}^X);\ Z_j^X = \varphi_{p,Y}^z(h_{j-1}^Y, \{h_i^X\}_{i=1}^{T_x});$$
$$h_j^Y = \varphi_{p,Y}^c(y_{j-1}, h_{j-1}^Y, Z_j^X);\ y_j = \varphi_{p,Y}^h(h_j^Y). \quad (1)$$

The dual task can be similarly formulated.

In the primal task, $\varphi_{p,X}^c$ serves to encode a source-language sentence x, without any condition. In the dual task, $\varphi_{d,X}^c$ is used to decode a target-language sentence x, conditioned on Z^Y. That is, given two dual tasks, the encoder of the primal task and the decoder of the dual task are highly correlated. They only differ in the conditions. Inspired by representation sharing (e.g., sharing the bottom layers of the neural network models for related tasks (Dong et al., 2015)), in this work, we propose to share components of the models of two tasks in dual form, e.g., forcing $\varphi_{p,X}^c = \varphi_{d,X}^c$ and $\varphi_{p,Y}^c = \varphi_{d,Y}^c$ for neural machine translation, and call such an approach *model-level dual learning*.

The strictly symmetric model architectures of a pair of dual tasks is good to have for model-level dual learning, but it is not a must-to-have. Take sentiment analysis as an example. The primal task is to classify whether a sentence is of positive sentiment or negative. Usually, the input sequence is first encoded to several hidden states by an LSTM and then fed into a few fully-connected layers to get the final classification decision. For the dual task, sentence generation with a given sentiment label, the sentiment label is first encoded into a hidden representation and then decoded into a sequence by an LSTM. In such asymmetric cases, the encoders of the primal task and the decoders of the dual task can be shared, which can be seen as a degenerated version of our proposed method by setting $\varphi_{p,X}^c = \varphi_{d,X}^c$ only.

Our main contributions can be summarized as follows:

(1) *Model Architecture* We re-formulate the current encoder-decoder framework as a combination of two conditional encoder and propose a unified architecture, *model-level dual learning*, that can handle two dual tasks simultaneously with the same set of parameters. We consider both the symmetric setting and the asymmetric setting, in which the primal and dual models are of the same/different architectures respectively.

(2) *Experimental Results* Our model is verified on two different tasks, neural machine translation and sentiment analysis. We achieve promising results: (1) On IWSLT14 German-to-English translation, we improve the BLEU score from 32.85 to 35.19, obtaining a new record (see Table 5); (2) On WMT14 English→German translation, we improve the BLEU score from 28.4 to 28.9 (see Table 3); (3) A series of state-of-the-art results on NIST Chinese-to-English are obtained (see Table 2). (4) With supervised data only, on IMDB sentiment classification dataset, we lower the error rate from 9.20% to 6.96% with our proposed framework (see Table 6).

The remaining part of the paper is organized as follows: The framework of model-level dual learning is introduced in Section 2. Section 3 and Section 4 show how to apply model level dual learning to neural machine translation and sentiment analysis. Section 5 discusses the combination of our method with dual inference (Xia et al., 2017a). Section 6 concludes this paper.

2. The Framework

We introduce the general framework of model-level dual learning in this section. Similar to the previous work on dual learning (He et al., 2016a; Xia et al., 2017b), we consider two spaces $\mathcal{X}$ and $\mathcal{Y}$ and two tasks in dual form: the primal task aims to learn a mapping $f : \mathcal{X} \mapsto \mathcal{Y}$, and the dual task learns a reverse mapping $g : \mathcal{Y} \mapsto \mathcal{X}$.

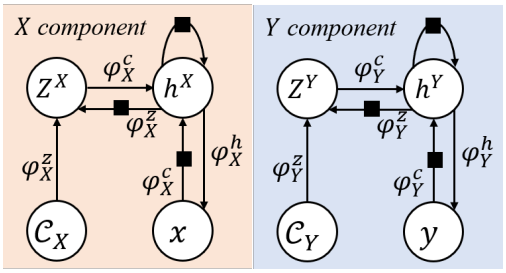
We consider two scenarios: the symmetric setting and the asymmetric setting. For the symmetric setting, the elements in $\mathcal{X}$ and $\mathcal{Y}$ are of the same format so that it is possible to use the same model architecture for the two mappings. For example, in NMT and Q&A, both $\mathcal{X}$ and $\mathcal{Y}$ are composed of natural language sentences and we can use LSTM to model both f and g. In the asymmetric setting, the objects in $\mathcal{X}$ and $\mathcal{Y}$ are of different formats and semantics, and thus the two mappings have different model architectures. For example, in sentiment analysis, $\mathcal{X}$ is the set of natural language sentences while $\mathcal{Y} = \{0, 1\}$ is the set of sentiment labels. The heterogeneity of $\mathcal{X}$ and $\mathcal{Y}$ forces one to use different model structures for the primal and dual tasks.

2.1. Symmetric Model-Level Dual Learning

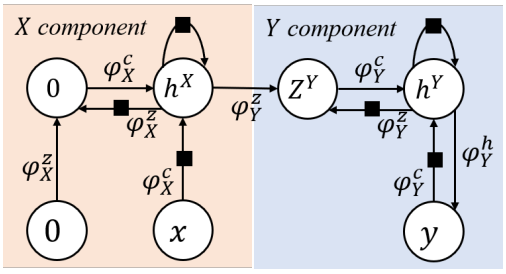
In the symmetric setting, the models f and g are made up of two parts: the X component φ_X and the Y component φ_Y. φ_X acts as both the encoder and decoder for space $\mathcal{X}$: in

the primal model f, it encodes a sample in $\mathcal{X}$ to get hidden representations; in the dual model g, it decodes some hidden representations and generates a sample in $\mathcal{X}$. Similarly, φ_Y acts as both the encoder and decoder for space $\mathcal{Y}$.

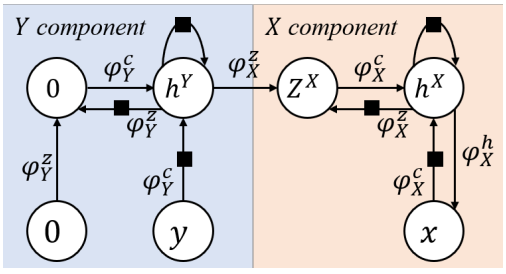
One may wonder why one component can act as both the encoder and the decoder, since compared with the encoder, the decoder of either $\mathcal{X}$ or $\mathcal{Y}$ typically takes additional information from the other space. For example, in neural machine translation, the encoder is a simple LSTM, but the decoder is an LSTM plus an attention module generating context from the source sentence (Bahdanau et al., 2015). This is easily solved via introducing the encoder a zero vector as such additional contexts (we name it as the "null context"). The detailed architecture is shown in Figure 2.



(a) X compotent and Y component



(b) Model for primal task, $\mathcal{C}_Y = h^X$



(c) Model for dual task, $\mathcal{C}_X = h^Y$

Figure 2. Architecture of symmetric model-level dual learning. The black square indicates an optional delay of a single time step. The unfold version can be found in Appendix B.

In Figure 2(a), the parameter φ_X of component X consists of three modules: (1) φ_X^c, used to combine x and Z^X;[1] (2) φ_X^z, used to combine h^X and $\mathcal{C}_X$; (3) φ_X^h, used to map the hidden states h^X to x.

Similarly, φ_Y of component Y contains φ_Y^c, φ_Y^z and φ_Y^h. $\mathcal{C}_X$ and $\mathcal{C}_Y$ are the context information used for decoding $\mathcal{X}$ and $\mathcal{Y}$, and they are both zero vectors when their corresponding components are used for encoding.

Now we show how the models f and g can be composed by φ_X and φ_Y. We take f as an example. It takes an $x \in \mathcal{X}$ as input and outputs a $y \in \mathcal{Y}$. According to Figure 2(b), the encoder and decoder of f are specified as follows.

(1) *The Encoder.* Set $\mathcal{C}_X$ to the null context, i.e., $\mathcal{C}_X = \{0\}$.[2] At step $i \in [T_x]$ where T_x is the length of x, preprocess $\mathcal{C}_X$ and obtain Z_i^X: $Z_i^X = \varphi_X^z(h_{i-1}^X, \mathcal{C}_X)$. φ_X^z is a function that sums up the elements in $\mathcal{C}_X$ with adaptive weights.[3] Then, calculate the hidden representation $h_i^X = \varphi_X^c(x, h_{i-1}^X, Z_i^X)$.[4] Eventually, we obtain a set of hidden representations $h^X = \{h_i^X\}_{i=1}^{T_x}$. The module φ_X^h in component X is not used while encoding $x \in \mathcal{X}$.

(2) *The Decoder.* Set $\mathcal{C}_Y$ to the hidden representations h^X obtained in the encoding phase. At step $j \in [T_y]$, where T_y is the length of y, preprocess $\mathcal{C}_Y$ with the information available at step j and obtain Z_j^Y: $Z_j^Y = \varphi_Y^z(h_{j-1}^Y, \mathcal{C}_Y)$. Calculate the hidden representation $h_j^Y = \varphi_Y^c(y_{<j}, h_{j-1}^Y, Z_j^Y)$. Then map h_j^Y to y_j by $y_j = \varphi_Y^h(h_j^Y)$. If y_j is the symbol indicating the end of a sentence, terminate the decoding procedure; otherwise, continue to generate words one by one.

The dual model g can be specified in a similar way as shown in Figure 2(c): it first uses φ_Y to encode a $y \in \mathcal{Y}$ and then uses φ_X to generate an $x \in \mathcal{X}$. More details can be found in Appendix C.

Note that φ_X and φ_Y can be specialized with different function classes, such as LSTM, CNN or self-attention (Lin et al., 2017) networks for neural machine translation.

2.2. Asymmetric Model-Level Dual Learning

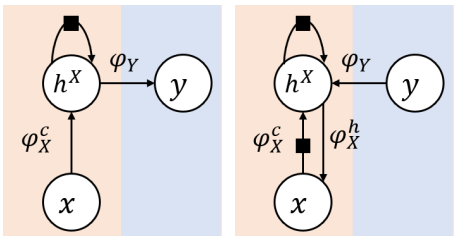
As aforementioned, in the asymmetric setting, $\mathcal{X}$ and $\mathcal{Y}$ are of different formats or semantics. Take sentiment analysis as an example. The primal task is to map a natural language sentence $x \in \mathcal{X}$ to a label $y \in \mathcal{Y}$. The dual task is to generate one or multiple sentences given the label information, such as generating a positive movie review or a negative comment to a restaurant.

[1] φ_X^c takes both Z^X and x as inputs. We plot this operator on the two edges connecting the two inputs.

[2] When the context is clear, we use 0 or $\{0\}$ alternatively.

[3] Such a function is widely adopted in NMT literature, which is exactly the attention module. That is, $\varphi_X^z(x, \{0\}) = 0$.

[4] Note that in the encoding phase, all words in x are available. At step i, φ_X^c and φ_X^z can consider either $x_{<i}$ (Bahdanau et al., 2015) or all the x_i's (Vaswani et al., 2017).



(a) primal model f (b) dual model g

Figure 3. Architecture of asymmetric model-level dual learning.

The architecture of the models for this setting is shown in Figure 3. The models f and g are also made up of two components, φ_X and φ_Y. Given a data pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$, for the primal task, for any $i \in [T_x]$,

$$h_i^X = \varphi_X^c(x_i, h_{i-1}^X),\ y = \varphi_Y(h_{T_x}^X). \tag{2}$$

In the above equation, φ_Y maps $h_{T_x}^X$ into a distribution over all the category labels using the category embedding matrix and the softmax operator. For the dual task, φ_Y first maps y, an one-hot vector/distribution, into a continuous vector space using the category embedding matrix, and then $\forall j \in [T_x]$, we have

$$h_j^X = \varphi_X^c(x_{j-1}, h_{j-1}^X, \varphi_Y(y)),\ x_j = \varphi_X^h(h_j^X), \tag{3}$$

where φ_X can also be implemented as LSTM, CNN, etc.

We would like to point out although the input/output spaces of φ_Y are different in the above two equations, φ_Y uses the same set of parameters, i.e., the category embedding matrix, and thus we use the same operator connecting h^X to y in Figure 3(a) and connecting y to h^X in Figure 3(b). More specifically, we set E_y as a $|\mathcal{Y}| \times d$ matrix where $|\mathcal{Y}|$ is the number of elements in $\mathcal{Y}$ and d is the dimension of hidden states. In Eqn.(2), $\varphi_Y(h_{T_x}^X) = \text{softmax}(E_y h_{T_x}^X)$; in Eqn.(3), $\varphi_Y(y) = E_y^T y$. Similarly, although φ_X^c takes different number of inputs in above equations, it uses the same set of parameters for the transformations.

2.3. Discussions

From the above descriptions, we can see that the primal model and dual model share the same set of parameters, but they organize the parameters in different ways. There are several consequences from this parameter sharing mechanism. First, we reduce the total number of parameters needed for the two tasks, achieving model compression from a very different perspective. Second, given a data pair (x, y), the parameters will be updated twice, one from the primal model (e.g., its gradient) and the other one from the dual model. To some extent, the parameters are trained more sufficiently than conventional supervised learning and data-level dual learning. Third, thanks to parameter sharing, we reduce the complexity of the two models and therefore they are likely to achieve better generation. A simple discussion about the generalization bound of model-level dual learning provided in Appendix D.

Model-level dual learning is a kind of *multi-task learning*. Both multi-task learning and model-level dual learning share parameters across tasks, but with different sharing mechanisms: Multi-task learning assumes tasks share the same input space and thus force their models to share low-level structures and parameters; model-level dual learning is based on the duality of tasks and organize the parameters of the two models in different directions.

There are also distinctions between our work and previous dual learning methods such as dual unsupervised learning (He et al., 2016a) and dual supervised learning (Xia et al., 2017b). As aforementioned, previous dual learning algorithms leverage the task duality on data level and change/enhance the loss functions of model training, either minimizing data reconstruction loss (He et al., 2016a; Lin et al., 2018) or ensure the consistency of data generation probability (Xia et al., 2017b; Wang et al., 2018). We bring the duality into model structures of the two tasks and do not change the loss functions. It is interesting to unify the two kinds of duality learning, i.e., combining model-level dual learning with data-level dual learning. We leave it to future work.

3. Application to Neural Machine Translation

Neural machine translation perfectly fits into the symmetric setting of the proposed dual learning framework. Therefore we apply the framework to this task in this section.

We choose the neural machine translation model *Transformer* (Vaswani et al., 2017) as our basic model architecture[5]. We first show how to adapt Transformer to our framework, and then conduct the empirical experiments to verify the effectiveness of our proposed method.

3.1. Model Adaption

In Figure 4, we show the mapping from different parts of Transformer to the submodules in our framework as introduced in Section 2.1. Both the encoder and decoder in Transformer consist of a list of stacked blocks. There are three types of layers in each block: (1) a self-attention layer, that sums up the hidden states from the bottom layer with adaptive weights; (2) an optional encoder-decoder attention layer, that outputs contextual information from the encoder. This layer only exists in the decoder side; (3) a feed-forward layer, that applies nonlinear transformation on the outputs of the self-attention and the encoder-decoder attention lay-

[5] *https://github.com/tensorflow/tensor2tensor*.

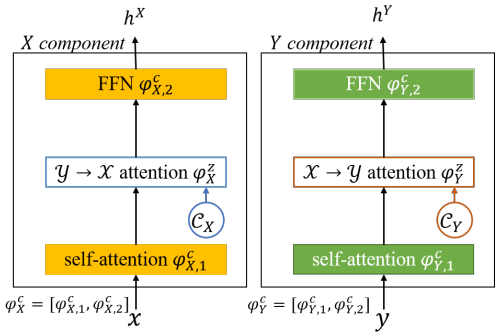


Figure 4. Model-level dual learning for NMT.

ers. Each layer is associated with a residual connection (He et al., 2016b) and a layer normalization operation (Ba et al., 2016).

We introduce the notations used in Figure 4. Both the two components φ^c_X and φ^c_Y contain two parts: the self-attention layer (denoted as $\varphi^c_{\cdot,1}$) and feed-forward layer with nonlinear transformations (denoted as $\varphi^c_{\cdot,2}$). Meanwhile φ^z_X and φ^z_Y are the inter attentions between encoder and decoder. φ^h_X and φ^h_Y are two softmax operations and for better readability, we omit them from Figure 4.

Now we take the translation from source language $\mathcal{X}$ to target language $\mathcal{Y}$ (i.e., the primal task) as an example to illustrate how to specify the model in our framework. In this task, the primal model works as follows:

$$\tilde{x} = \varphi^c_{X,1}(x),\ h^X = \varphi^c_{X,2}(\tilde{x} + \varphi^z_X(\tilde{x}, \mathcal{C}_X)), \quad (4)$$

where $\mathcal{C}_X = \{0\}$, $h^X = \{h^X_i\}_{i=1}^{T_x}$. Considering the cross-space attention $\mathcal{Y} \to \mathcal{X}$ is actually a linear combination of the elements in set $\mathcal{C}_X$, we know that $\varphi^z_X(\tilde{x}, 0) = 0$.

In different blocks, the x in Eqn.(4) is specified in different ways: (i) In the first block, it is the embeddings of individual words in the source input sentence; (ii) in the $i(>1)$th block, it is the hidden state h^X outputted by the $(i-1)$th layer.

The Y component works as follows. To decode the tth word y_t, given the previously generated words $y_{<t}$, we have

$$\tilde{y}_{<t} = \varphi^c_{Y,1}(y_{<t}),\ h^Y_t = \varphi^c_{Y,2}(\tilde{y}_{<t} + \varphi^z_Y(\tilde{y}_{<t}, \mathcal{C}_Y)), \quad (5)$$

where $\mathcal{C}_Y = h^X$, which is the output of the last block of the X component, and further, $y_t = \varphi^h_Y(h^Y_t)$. The input $y_{<t}$ is specified in a similar way as that for the X component.

In the primal model, the X component and Y component work like the encoder and decoder in (Vaswani et al., 2017). The dual model can be specified similarly by first using the Y component and then the X component.

3.2. Settings

Datasets We choose three widely used neural machine translation tasks: IWSLT 2014 German↔English (briefly, IWSLT De↔En), LDC Chinese↔English (briefly, Zh↔En) and WMT14 English↔German (briefly, WMT En↔De) as our testbeds. (1) For IWSLT De↔En, we use the data extracted from IWSLT 2014 evaluation campaign (Cettolo et al., 2014), which consists of $153k$/$7k$/$7k$ sentence pairs as training/validation/test sets. (2) For Zh↔En, we extract 1.25M bilingual sentences pairs from LDC dataset, which is the same as (Xia et al., 2017d). NIST 2003 acts as the validation set, and NIST 2004/2005/2006/2008/2012 as test sets[6]. (3) For WMT En↔De, same as previous works (Jean et al., 2015; Vaswani et al., 2017; Gehring et al., 2017), we use the training set consisting of roughly 4.5M sentence pairs. We use *newstest13* and *newstest14* as the validation and test sets respectively. Since subword-level translation is proven to outperform word-level translation on several tasks, we apply the BPE (Sennrich et al., 2016) techniques to Zh↔En translation tasks with $25k$ merge operations and apply the techniques proposed in (Wu et al., 2016) to split the words into wordpieces for the other two tasks. We set source and target vocabulary sizes of the three tasks as $(25k, 25k)$, $(37k, 22k)$ and $(32k, 32k)$ respectively. Out-of-vocabulary words are replaced with a special token "UNK".

Model Configurations For all experiments, both the encoder and decoder contain six blocks. We use the *transformer_small* setting for IWSLT De↔En, whose word embedding dimension, hidden size and feed-forward filter size are 256, 256 and 1024 respectively, and *transformer_big* setting for Zh↔En and WMT En↔De, where the three corresponding dimensions are 1024, 1024 and 4096.[7] The residual dropouts of the three tasks are 0.1, 0.3 and 0.3 respectively. We use weight tying (Press & Wolf, 2016) for the IWSLT De↔En and WMT En↔De translation, i.e., the source and target word embeddings as well as the softmax matrix are shared.

Training and Inference Following (Vaswani et al., 2017), we use Adam (Kingma & Ba, 2014) as the optimizer, with initial learning rates 0.0002, $\beta_1 = 0.9$ and $\beta_2 = 0.98$. Each mini-batch in all tasks contains around 4096 tokens. All the models are trained using NVIDIA Tesla M40 GPU. For training our dual learning models, the training cost is 2 GPUs and 1 day for IWSLT De↔En, 8 GPUs and 1 day for Zh↔En, 8 GPUs and 7 days for WMT En↔De. At

[6] It is a common practice to report results on multiple test sets for Chinese→English translation task (Wu et al., 2017; Shen et al., 2016; Su et al., 2018).

[7] We carry out several preliminary experiments to choose the model structures. On the relatively small dataset IWSLT De↔En, we find that *transformer_base* and *transformer_big* cannot get reasonable BLEUs due to overfitting. On the other two tasks, *transformer_big* performs the best.

inference phase, we use beam search with beam sizes 6, 6, 4 for the three tasks. The translation quality is evaluated by tokenized BLEU (Papineni et al., 2002). Following the common practice in NMT, we use case-sensitive BLEU to WMT En↔De and case-insensitive BLEU to other tasks.

Furthermore, we implement dual supervised learning (Xia et al., 2017b) based on Transformer as a baseline for comparison, which is briefly denoted as *DSL*.

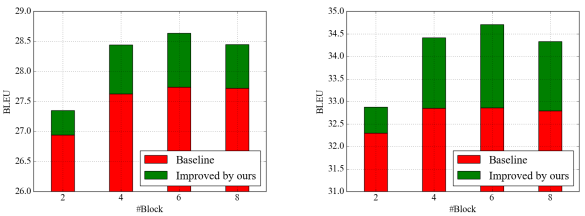
3.3. Results

The results on IWSLT De↔En are shown in Table 1, from which we have several observations. First, we achieve a new state-of-the-art BLEU score, 34.71, for De→En task, outperforming all existing works such as (Wang et al., 2018) and (Edunov et al., 2018). Second, compared with the Transformer baseline on both the primal task De→En and the dual task En→De, our approach can boost the BLEU scores by 1.85 and 0.90 points respectively. It is noteworthy that our basic model structure is exactly the same with the baseline for both the primal and dual tasks. This clearly shows the benefit of pushing duality into model architectures. Finally, compared with DSL, we achieved 1.13 and 0.73 improvements, demonstrating that the duality constraint is better imposed on the model level than data level.

Table 1. BLEU scores on IWSTL14 De→En. We do not find reasonable numbers for IWSLT En→De translation task since most research works focus on IWSLT De→En.

Existing Results on IWSLT De→En	
GRU + Dual Learning (Wang et al., 2018)	32.05
GRU + Dual Transfer Learning (Wang et al., 2018)	32.35
CNN + reinforcement learning (Edunov et al., 2018)	32.85

Model	**De→En**	**En→De**
Transformer	32.86	27.74
DSL	33.58	27.91
Ours	**34.71**	**28.64**



(a) En→De (b) De→En

Figure 5. BLEUs w.r.t different number of blocks.

To investigate how our algorithm works with respect to different model complexities, we train a group of models with 2/4/6/8 blocks[8] respectively. We visualize their test BLEUs in Figure 5. We have several observations. First, for all the settings of different numbers of blocks, our work consistently improves over baseline for both the primal and dual tasks. Second, when the number of blocks increases from 2 to 6, the improvement brought by our method (indicated by the green areas) becomes more significant on both translation directions. Checking the results of the baseline method (the red areas), we can see that although deeper models have more capacities for better results, they might suffer from overfitting, especially on the small scale task like IWSLT De↔En. Our method mitigates this issue by pushing duality into model architectures, consequently unleashing the potential of deeper models. Third, even for the deepest models with 8 blocks, our method also brings accuracy gain, although not as significant as shallower models. In this case, to further improve generalization, it is worthy to combine our method with other regularization method, e.g., dual supervised learning (Xia et al., 2017b).

The translation results on the Zh↔En task are summarized in Table 2. Again, we achieve state-of-the-art results, outperforming quite strong baseline using more complicated model structures (Su et al., 2018; Wu et al., 2017) and larger amount of supervised data (Xia et al., 2017b). Here part of our gain comes from the superior *transformer_big* model, on top of which we can still increase the BLEU scores by 1.21 points on average. Our method also beats DSL on all the test datasets, furthering demonstrating the effectiveness of model-level dual learning.

For En→Zh translation, we again achieve good improvements. Compared with the baseline, we can achieve 0.69 BLEU improvement on average.

At last, we show the results on WMT En↔De translation in Table 3. To the best of our knowledge, Transformer achieves the best BLEU score on this dataset, significantly outperforming the other baselines. Given such a strong baseline, we are still able to increase the BLEU scores by 0.5 point for both tasks, although the improvements are not as significant as the other tasks. In addition, for the dual task De→En translation, we also obtain 0.5 point improvement for BLEU score.

Overall speaking, the improvements on neural machine translation experiments demonstrate the success of applying model-level dual learning to sequence-to-sequence tasks.

Discussion (1) One more point we want to mention is that, for neural machine translation, although it is not our main motivation, we can reduce the model size by a non-trivial margin via parameter sharing. For the IWSLT 2014 German↔English translation tasks with *transformer_small* configurations, we need two separated models, resulting in

[8] Keep in mind that each block contains 3 layers.

Table 2. Translation results of Zh↔En. Blank tabular means that the corresponding results are not reported.

Zh→En	**NIST 04**	**NIST 05**	**NIST 06**	**NIST 08**	**NIST 12**
MRT (Shen et al., 2016)	41.37	38.81	29.23	-	-
VRNMT (Su et al., 2018)	41.07	36.82	36.72	-	-
SD-NMT (Wu et al., 2017)	-	39.38	41.81	33.06	31.43
GRU+DSL (Xia et al., 2017b)	-	-	-	33.59	32.00
Transformer	42.62	43.13	41.41	33.43	32.75
DSL	42.90	43.21	41.99	34.41	32.93
Ours	**43.38**	**44.16**	**42.60**	**35.05**	**34.19**

En→Zh	**NIST04**	**NIST05**	**NIST06**	**NIST08**	**NIST12**
Bi-Attn (Cheng et al., 2016)	16.98	15.70	16.25	13.80	-
GRU+DSL (Xia et al., 2017b)	-	-	-	15.87	16.10
Transformer	23.24	21.76	21.67	17.23	15.76
DSL	23.62	22.22	22.31	17.79	16.61
Ours	**24.23**	**22.46**	**21.80**	**18.06**	**16.54**

Table 3. Translation Results of WMT14 En↔De.

	En→De	**De→En**
GNMT (Wu et al., 2016)	24.61	-
CNN (Gehring et al., 2017)	25.16	29.61
Model	**En→De**	**De→En**
Transformer	28.4	31.4
Ours	**28.9**	**31.9**

$38M$ parameters. However, if we use our proposed model-level dual learning, we only need one model with $20M$ parameters, which saves about 47% parameters. For NIST Chinese↔English translations based on *transformer_big* models, the aforementioned two numbers turn to $556M$ and $329M$ respectively. (2) (Firat et al., 2016a;b) also propose parameter sharing in NMT but in a very different way from our work: their decoders only serve for decoding and encoders only for encoding, no matter for which translation task; in comparison, the decoder and encoder play two roles in our work, e.g., the decoder serves for decoding in the primal task and encoding in the dual task. Furthermore, the motivation of weight sharing in those papers largely lies in improving the performance for low/zero resource translation in the setting of multi-lingual translation. As a comparison, we focus on the typical setting that only bilingual training corpus is available.

4. Application to Sentiment Analysis

In this section, we apply model-level dual learning to sentiment analysis, in which the primal task is sentiment classification that aims to justify whether a natural language sentence is of positive sentiment or negative. The dual task is language modeling of a sentence conditioned on a given positive or negative sentiment label. In this application, $\mathcal{X}$ is the space of natural sentences, and $\mathcal{Y}$ is the space of sentiment labels. Clearly, the objects in the two spaces are of different formats, and thus this is an asymmetric setting.

4.1. Model Adaption

We choose the classical LSTM network as the basic building block of both the primal and dual tasks. That is, The $\varphi_X^c(x_{t-1}, h_{t-1}, y)$ in Section 2.2 is specified as follows:

$$
\begin{aligned}
&[z_t^i, z_t^o, z_t^f, z_t^c] = W_a x_{t-1} + U_a h_{t-1} + E_y y + b_a; \\
&i_t = \sigma(z_t^i);\ o_t = \sigma(z_t^o);\ f_t = \sigma(z_t^f);\ \tilde{C}_t = \tanh(z_t^c); \\
&C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t;\ h_t = o_t \odot \tanh(C_t),
\end{aligned} \tag{6}
$$

where the W_a, U_a, E_y and b_a are the parameters to be learned, $\odot$ represents element-wise product and $\sigma(\cdot)$ is the sigmoid function. In the primal task, $y = 0$ and E_y is used as the output embedding matrix to predict the sentiment label. In the dual task, y is replaced with the corresponding category label. Thus, W_a, U_a, E_y and b_a are shared across the primal task and dual task.

4.2. Settings

Dataset We use the benchmark movie review dataset IMDB (Maas et al., 2011) for sentiment analysis. The corpus consists of $25k$ training samples and $25k$ test samples, as well as another $50k$ unlabeled sentences. We in this paper focus on using labeled data only. For validation purpose, we randomly split 3750 samples from the training set as the validation set. The vocabulary consists of the most frequent $10k$ words. Out-of-vocabulary words are replaced with the token "UNK".

Model Configurations Following (Xia et al., 2017b), we set the embedding dimension (both word embedding and sentiment embedding) and the LSTM hidden layer size as 500 and 1024 respectively. The dropout rate is fixed as 0.5 for both embeddings and softmax.

Table 4. Results of sentiment analysis on IMDB dataset (supervised data only). Existing results include [1] (Dai & Le, 2015) [2] (Johnson & Zhang, 2015) [3] (Johnson & Zhang, 2016).

Previous Works	**Error Rate (%)**
Standard LSTM [1]	10
oh-CNN [2]	8.39
oh-2LSTMp [3]	8.14

Model	**Error Rate (%)**	**Perplexity**
LSTM	10.10	59.19
DSL	9.20	58.78
Ours	**7.41**	**55.59**

Training and Evaluation We use Adadelta (Zeiler, 2012) to optimize our proposed model. When the validation loss does not decrease, we use plane SGD to finetune the model with learning rate 0.05 till convergence, indicated by that the accuracy on the validation set ceases increasing. The whole training process takes three days on a single Titan XP GPU. Following (Xia et al., 2017b), we evaluate the primal task (i.e., sentiment classification) by error rate, and the dual task (i.e., sentence modeling with conditional sentiment) by conditional perplexity, which is defined as follows: perplexity $= \exp\big(-\sum_{i=1}^{n}\log P(x^{(i)}|y^{(i)})/\sum_{i=1}^{n}N_i\big)$, where n is the number of sentences in the test set, $x^{(i)}$ and $y^{(i)}$ are the ith data and label, and N_i is the length of $x^{(i)}$. The smaller the perplexity is, the better the model is.

4.3. Results

Our results on sentiment analysis are summarized in Table 4. We can see in the primal task, our proposed method can reduce the classification error rate by 2.69 point, significantly outperforming standard LSTM. Also, we can reduce the error rate by 1.79 points compared with DSL. In the dual task, we can achieve 55.59 test perplexity, which is 3.19 points lower than the result of DSL. These results validate the effectiveness of model-level dual learning under the asymmetric setting.

5. Combination with Dual Inference

Dual inference (Xia et al., 2017a) is proposed to leverage the duality of two tasks to mutually enhance the inference quality. Mathematically, given the primal model $f : \mathcal{X} \mapsto \mathcal{Y}$ and the dual model $g : \mathcal{Y} \mapsto \mathcal{X}$, dual inference makes predictions for the two tasks as follows:

$$\begin{aligned} y^* &= \text{argmin}_{y\in\mathcal{Y}}\alpha\ell_f(x,y) + (1-\alpha)\ell_g(x,y), \\ x^* &= \text{argmin}_{x\in\mathcal{X}}\beta\ell_g(x,y) + (1-\beta)\ell_f(x,y), \end{aligned} \tag{7}$$

where ℓ_f and ℓ_g are the loss functions of the primal and dual tasks, and α and β are trade-off hyperparameters, selected by the validation performances. Since our approach represents two models using the same set of parameters and jointly train the two models, it is naturally to combine our learnt models with dual inference.

Table 5. Dual inference results on IWSLT De↔En. "DI" stands for "dual inference". The third row represents the performance of Transformer with DI or "standard" inference (i.e., without DI). The fourth row represents the performance of model-level dual learning (also based on Transformer) w/o dual inference.

	De→En		**En→De**	
Model	**Standard**	**DI**	**Standard**	**DI**
Transformer	32.86	33.52	27.74	27.84
Ours	34.71	**35.19**	28.64	**28.83**

We carry out experiments on IWSLT De↔En translation and sentiment classification to check the effectiveness of the combination of model-level dual learning and dual inference. The experimental results of IWSLT De↔En are in Table 5. We can see that dual inference can boost our method by 0.48 point (from 34.71 to 35.19) for De→En translation and by 0.19 point (from 28.64 to 28.83) for En→De translation, which shows that our proposed dual learning can still be improved by further processing.

Table 6. Error rates on IMDB classification w/o dual inference.

Model	**Standard**	**DI**
$\mathcal{M}_{\text{w2v}}$ (Xia et al., 2017a)	10.10	8.31
Ours	**7.41**	**6.96**

Table 6 shows the dual inference results on IMDB sentiment classification. $\mathcal{M}_{\text{w2v}}$ represents the standard LSTM classifier. As shown in the table, by applying dual inference, we obtain 0.45 classification accuracy improvement, which demonstrates that dual inference also works for this asymmetric setting.

6. Conclusions and Future Work

In this work, we proposed a learning framework, model-level dual learning, which leverages task duality to redesign the model architectures for both the primal and dual task. We demonstrated the effectiveness of the proposed framework on neural machine translation and sentiment analysis.

For future work, there are multiple directions to explore. First, it is interesting to apply the framework to more scenarios such as image processing and speech processing. Second, it is worthy to combine model-level dual learning with data-level dual learning, which might achieve even better performance for real-world problems. Third, we will study how to leverage task duality to learn from scratch without supervised data, e.g., machine translation without parallel data (Artetxe et al., 2018; Lample et al., 2018).

Acknowledgements

We thank the anonymous reviewers for their helpful comments. This work was done when Yingce Xia was an intern at Microsoft Research.

References

Artetxe, M., Labaka, G., Agirre, E., and Cho, K. Unsupervised neural machine translation. *International Conference on Learning Representations*, 2018.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., and Federico, M. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, 2014.

Cheng, Y., Shen, S., He, Z., He, W., Wu, H., Sun, M., and Liu, Y. Agreement-based joint training for bidirectional attention-based neural machine translation. *IJCAI*, 2016.

Dai, A. M. and Le, Q. V. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems 28*, 2015.

Dong, D., Wu, H., He, W., Yu, D., and Wang, H. Multi-task learning for multiple language translation. In *ACL*, volume 1, pp. 1723–1732, 2015.

Edunov, S., Ott, M., Auli, M., Grangier, D., and Ranzato, M. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of NAACL-HLT 2018*, 2018.

Firat, O., Cho, K., and Bengio, Y. Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv preprint arXiv:1601.01073*, 2016a.

Firat, O., Sankaran, B., Al-Onaizan, Y., Vural, F. T. Y., and Cho, K. Zero-resource translation with multi-lingual neural machine translation. *arXiv preprint arXiv:1606.04164*, 2016b.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. Convolutional sequence to sequence learning. *ICML*, 2017.

He, D., Xia, Y., Qin, T., Wang, L., Yu, N., Liu, T., and Ma, W.-Y. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pp. 820–828, 2016a.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016b.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jean, S., Cho, K., Memisevic, R., and Bengio, Y. On using very large target vocabulary for neural machine translation. *ACL*, 2015.

Johnson, R. and Zhang, T. Effective use of word order for text categorization with convolutional neural networks. *NAACL HLT*, 2015.

Johnson, R. and Zhang, T. Supervised and semi-supervised text categorization using lstm for region embeddings. In *International Conference on Machine Learning*, pp. 526–534, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. arxiv. org, 2014.

Lample, G., Conneau, A., Denoyer, L., and Ranzato, M. Unsupervised machine translation using monolingual corpora only. *International Conference on Learning Representations*, 2018.

Lin, J., Xia, Y., Qin, T., Chen, Z., and Liu, T.-Y. Conditional image-to-image translation. In *CVPR*, 2018.

Lin, Z., Feng, M., Santos, C. N. d., Yu, M., Xiang, B., Zhou, B., and Bengio, Y. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. Multi-task sequence to sequence learning. *ICLR*, 2016.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pp. 142–150. Association for Computational Linguistics, 2011.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10): 1345–1359, 2010.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Press, O. and Wolf, L. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*, 2016.

Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. *ACL*, 2016.

Shen, S., Cheng, Y., He, Z., He, W., Wu, H., Sun, M., and Liu, Y. Minimum risk training for neural machine translation. *ACL*, 2016.

Su, J., Wu, S., Xiong, D., Lu, Y., Han, X., and Zhang, B. Variational recurrent neural machine translation. In *AAAI*, 2018.

Tang, D., Duan, N., Qin, T., and Zhou, M. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 6000–6010, 2017.

Wang, Y., Xia, Y., Zhao, L., Bian, J., Qin, T., Liu, G., and Liu, T. Dual transfer learning for neural machine translation with marginal distribution regularization. In *AAAI*, 2018.

Wu, S., Zhang, D., Yang, N., Li, M., and Zhou, M. Sequence-to-dependency neural machine translation. In *ACL*, 2017.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Xia, Y., Bian, J., Qin, T., Yu, N., and Liu, T.-Y. Dual inference for machine learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3112–3118, 2017a.

Xia, Y., Qin, T., Chen, W., Bian, J., Yu, N., and Liu, T.-Y. Dual supervised learning. *ICML*, 2017b.

Xia, Y., Tian, F., Qin, T., Yu, N., and Liu, T.-Y. Sequence generation with target attention. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 816–831. Springer, 2017c.

Xia, Y., Tian, F., Wu, L., Lin, J., Qin, T., Yu, N., and Liu, T.-Y. Deliberation networks: Sequence generation beyond one-pass decoding. In *Advances in Neural Information Processing Systems*, pp. 1782–1792, 2017d.

Yi, Z., Zhang, H., Tan, P., and Gong, M. Dualgan: Unsupervised dual learning for image-to-image translation. *ICCV*, 2017.

Zeiler, M. D. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Zhang, Y. and Yang, Q. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.