Learning Robust, Transferable Sentence Representations for Text Classification

Wasi Uddin Ahmad[1], **Xueying Bai**[2], **Nanyun Peng**[3], **Kai-Wei Chang**[1]
[1]University of California, Los Angeles, [2]Stony Brook University
[3]University of Southern California
{wasiahmad,kwchang}@cs.ucla.edu
xubai@cs.stonybrook.edu,npeng@isi.edu

Abstract

Despite deep recurrent neural networks (RNNs) demonstrate strong performance in text classification, training RNN models are often expensive and requires an extensive collection of annotated data which may not be available. To overcome the data limitation issue, existing approaches leverage either pre-trained word embedding or sentence representation to lift the burden of training RNNs from scratch. In this paper, we show that jointly learning sentence representations from multiple text classification tasks and combining them with pre-trained word-level and sentence level encoders result in robust sentence representations that are useful for transfer learning. Extensive experiments and analyses using a wide range of transfer and linguistic tasks endorse the effectiveness of our approach.

1 Introduction

Recent advances in deep neural networks have demonstrated the capability to build highly accurate models by training on vast amounts of data. The efficiency of these techniques comes from their ability to learn an encoder to convert raw inputs into useful continuous feature representations effectively. These successes primarily credit to the availability of ample resources, such as an extensive collection of training data. However, collecting a sufficient amount of manually annotated data is not always feasible, especially for domains requiring expert annotators.

While human annotated data is limited, there are abundant resources that can be used to lift the burden of learning representations from scratch and thus subsidize the requirement of having a large amount of training data. In the context of modeling natural languages, many successful stories showed that learned representations in both word and sentence levels are transferable to other tasks. These pre-trained representations enable us to model many natural language processing (NLP) tasks such as text classification (Bailey & Chopra, 2018) and named entity recognition (Cherry & Guo, 2015) with only a few thousands of examples.

At the word level, pre-trained word embeddings (Mikolov et al., 2013; Pennington et al., 2014) encode each word into a continuous vector representation, have been widely used in many applications (Seo et al., 2017; Lee et al., 2017; Venugopalan et al., 2017; Teney et al., 2017). A few recent methods propose to construct *contextualized word vectors* to address the issue that the meaning of a word should be context-dependent. For example, Peters et al. (2018) leveraged a large unannotated corpus to train such contextualized word vectors by feeding word sequences into a deep recurrent neural network (RNNs) and generating representations based on the hidden states of the RNNs correspond to the respective words. This results in impressive performances in many NLP applications (Lee et al., 2017; Peters et al., 2017; He et al., 2017).

At the sentence level, Conneau et al. (2017) showed that an LSTM-based sentence encoder (Hochreiter & Schmidhuber, 1997) trained on an annotated corpus for natural language inference (NLI) (Bowman et al., 2015) can capture useful features that are transferable to a wide range of text classification tasks. A few follow-up studies (Subramanian et al., 2018; Logeswaran & Lee, 2018; Cer et al., 2018) extended the approach by leveraging *large-scale* data and studied how to learn better transferable sentence representations.

However, all the existing approaches considered training word or sentence level representations from scratch. In contrast, we argue that by leveraging pre-trained embeddings/encoders and employing multiple large-scale supervised text classification datasets, we can learn more robust and transferable sentence representations. The primary research question that we address is how to build robust and transferable representations for sentence classification. The key challenges are two folds: 1) how to transfer only salient features by distinguishing generic information from task-specific information when learning an encoder and 2) how to combine representations at word and sentence levels to build strong transferable representations for sentence classification.

To address the first challenge, we propose to leverage multi-task learning (MTL) to jointly train sentence encoders on three large-scale text classification corpora, which cover a variety of domains and two language classification tasks – textual entailment and question paraphrasing. We exploit an MTL architecture that learns to separate generic representations from task-specific representations using adversarial training. While generic representations capture language-specific information, i.e., language structure, syntax, and semantics that are useful uniformly across a variety of language tasks, the task-specific representations encode domain knowledge that is helpful if the source and transfer tasks are homogeneous. Our experimental results show that when the shared and task-specific encoders are combined, they become more effective and applicable to a wide range of transfer tasks.

Besides, we combine our MTL-based sentence encoders with another existing sentence encoder (Subramanian et al., 2018) trained with different learning signals, and contextualized word vectors (Peters et al., 2018), to build a more robust and transferable sentence encoder. We evaluate our encoder on 15 transfer (Conneau & Kiela, 2018) and 10 linguistic probing (Conneau et al., 2018) tasks. Experimental results demonstrate that our proposed sentence encoder better captures linguistic information and provides a significant improvement over existing transfer learning approaches. We have made our encoders publicly available.[1]

2 Related Work

Our work is closely related to sentence representation learning, multi-task learning, and transfer learning and we briefly review each of these areas in this section.

• **Sentence Representations Learning.** Training neural networks to form useful sentence representations has become a core component in many machine learning models. Learning distributional sentence representations such that they capture the syntactic and semantic regularities has been proposed. These approaches range from models that compose of word embeddings (Le & Mikolov, 2014; Arora et al., 2017; Wieting et al., 2016) to models with complex network architectures (Zhao et al., 2015; Wang & Jiang, 2016; Liu et al., 2016c; Lin et al., 2017). Unsupervised approaches are also proposed in literature by utilizing a large collection of unlabeled text corpora to learn distributional sentence representations. For example, Kiros et al. (2015) revised the skip-gram model (Mikolov et al., 2013) to learn a generic sentence encoder, called SkipThought that is further improved by using layer normalization (Ba et al., 2016). Among other closely related works, the technique proposed by (Hill et al., 2016) fell short to SkipThought while (Logeswaran & Lee, 2018) showed improvement over skip-thought vectors.

Unlike word embeddings, learning sentence representations in an unsupervised fashion lack the reasoning about semantic relationships between sentences. To this end, Conneau et al. (2017) proposed to train a universal sentence encoder in the form of a bidirectional LSTM using the *supervised* natural language inference data, outperforming unsupervised approaches like SkipThought. Subramanian et al. (2018) propose to build general purpose sentence encoder by learning from a joint objective of classification, machine translation, parse tree generation and unsupervised skip-thought tasks. Compared to their approach, we propose to utilize multiple text classification datasets by leveraging a multi-task learning approach and combine them with existing contextualized word vectors (McCann et al., 2017; Peters et al., 2018) to learn robust and transferable sentence representations. Recent works (Cer et al., 2018; Perone et al., 2018) explored RNN free sentence encoders and evaluated sentence representations learning methods by using a variety of downstream and linguistic tasks.

• **Multi-task Learning (MTL).** Multi-task learning has been successfully used in a wide-range of natural language processing applications, including text classification (Liu et al., 2017), machine translation (Luong et al., 2016), sequence labeling (Rei, 2017), sequence tagging (Peng & Dredze,

[1] https://github.com/wasiahmad/transferable_sent2vec

2017), dependency parsing (Peng et al., 2017) etc. Recent works (Liu et al., 2016b; Zhang et al., 2017b; Liu et al., 2016a) proposed multi-task learning architectures with different methods of sharing information across the participant tasks. To facilitate scaling and transferring when a large number of tasks are involved, Zhang et al. (2017a) proposed to embed labels by considering semantic correlations among tasks. To investigate how much transferable an end-to-end neural network architectures are for NLP applications, Mou et al. (2016) propose to use multi-task learning on sentence classification tasks. In contrast to these prior work, we aim to learn a universal sentence encoders via multi-task learning that are transferable to a wide range of heterogeneous tasks.

• **Transfer Learning.** Transfer learning stores the knowledge gained from solving source tasks (usually with abundant annotated data), and apply it to other tasks (usually suffer from insufficient annotated data to train complex models) to combat the inadequate supervision problem. It has become prevalent in many computer vision applications (Sharif Razavian et al., 2014; Antol et al., 2015) where image features were trained on ImageNet (Deng et al., 2009), and applications where word vectors (Pennington et al., 2014; Mikolov et al., 2013) were trained on large unlabeled corpora. Despite the benefits of using pre-trained word embeddings, many NLP applications still suffer from lacking high quality generic sentence representations that can help unseen tasks. In this work, we combine sentence representations learned using MTL and contextualized word vectors to obtain more robust sentence representations that transfer better.

3 Sentence Representations Learning

Our goal is to leverage available text corpora and existing sentence and word encoders to build a universal sentence encoder. In the following, we first define the sentence encoder and then describe a multi-task learning approach that learns sentence representations jointly on multiple text classification tasks. Then we discuss how to combine the learned sentence representations with the existing sentence and contextualized word vectors.

3.1 Sentence Encoder

A typical text classification model consists of two parts: a representation learning component, also known as encoders that convert input text sequences into fixed-size vectors, and a classifier component that takes the vector representations and predicts the final class labels. The encoder is usually realized by a high complexity neural network architecture and requires a large amount of data to train, as opposed to the classifier which is generally simple (e.g., a linear model). When enough training examples are provided, the encoder and the classifier can be trained jointly from scratch in an end-to-end fashion.[2] However, when data is insufficient, this approach is unfeasible. Instead, we can pre-train the encoder on other tasks (a.k.a source tasks) and transfer the learned encoder to the target task. In this case, we only require a few labeled examples to train the low-complexity classifier on top of the pre-trained encoder. We discuss how to build the pre-train encoder in below.

We follow (Conneau et al., 2017) to build a transferable encoder based on an one layer bidirectional LSTM with max pooling (BiLSTM-max). Formally, given a sentence with T words, $[w_1, w_2, ..., w_T]$, the encoder first runs two LSTM models on input text from both directions.

$$\overrightarrow{h}_t = LSTM(\overrightarrow{h}_{t-1}, w_t),\ \overleftarrow{h}_t = LSTM(\overleftarrow{h}_{t+1}, w_t) \tag{1}$$

and $h_t = [\overrightarrow{h}_t, \overleftarrow{h}_t] \in R^{2d}$ is the t-th hidden vectors in BiLSTM, d is the dimensionality of the LSTM hidden units. To form a fixed-size vector representation of variable length sentences, the maximum value is selected over each dimension of the hidden units:

$$s_j = \max_{t \in [1,...,T]}\ h_{j,t},\ j = 1, ..., d, \tag{2}$$

where s_j is the j-th element of the sentence embedding s.

For some transfer tasks (e.g., textural entailment and similarity measuring), the goal is to predict the relationship between two sentences. Therefore, the input involves two sentences (s_1, s_2). We generate the representation of input instances by $[s_1, s_2, s_1 - s_2, s_1 \odot s_2]$ where $\odot$ denotes the element-wise multiplication, and $[\cdot, \cdot]$ denotes vector concatenation.

[2] In this case, the classifier is the last layer in the network architecture.

Multi-task learning. Multi-task learning was shown efficient in many text classification tasks. However, its effectiveness in learning transferable sentence representations is comparably less studied. In this paper, we investigate the utility offered by several large-scale text classification tasks. We show that learning signals from various text classification tasks results in robust and transferable sentence representations. Inspired by (Liu et al., 2017), we study two variants of shared-private (SP) multi-task learning (MTL) frameworks.The shared-private MTL framework maintains private and shared encoders with their own task-specific layers to encourage task-specific and generic features being learned by the private and shared encoders, respectively. In this study, we design one private BiLSTM-max sentence encoder for each task, and one shared BiLSTM-max encoder for all the tasks to capture task-dependent and generic features, respectively. Sentence embeddings produced by private and shared encoders are concatenated to form the final sentence representations. In this way, the shared encoder provides task-independent information, while the private encoders are helpful when the target task is proximity to some source tasks. Formally, for any sentence in a given task k, its shared representation s_s^k and private representation s_p^k can be computed using Eq. (1) – (2), and they are concatenated to construct the sentence embedding: $s^k = [s_s^k, s_p^k]$.

Adversarial Training. Ideally, we want the private encoders to learn only task-specific features, and the shared encoder to learn generic features. To achieve this goal, we adopt the adversarial training strategy proposed by Liu et al. (2017) to introduce a discriminator on top of the shared BiLSTM-max sentence encoder. The goal of the discriminator, D is to identify which task an encoded sentence s^k comes from, and the adversarial training requires the shared sentence encoder to generate representations that can "fool" the discriminator. In this way, the shared encoder is forced not to carry task-related information. The discriminator is defined as,

$$D(s^k) = \textit{softmax}(Ws^k + b),$$

where $W \in R^{d \times d}$ and $b \in R^d$ are model parameters. Optimizing the adversarial loss,

$$L_{adv} = \min_{\theta_E} \left(\max_{\theta_D} \Big(\sum_{k=1}^{K} \sum_{i=1}^{N_k} d_i^k \log[D(E(s))] \Big) \right)$$

has two competing goals: the discriminator tries to maximize the classification accuracy (inside the parentheses), and the sentence encoder tries to confuse it (and thus minimize the classification accuracy). E and D represents the shared sentence encoder and the discriminator respectively and θ_E and θ_D are the model parameters of E and D. d_i^k denotes the ground-truth label indicating the type of the current task. To encourage the shared and private encoders to capture different aspects of the sentences, the following term is added.

$$L_{diff} = \sum\nolimits_{k=1}^{K} \left\| {H_s^k}^\top H_p^k \right\|_F^2$$

where $\|\cdot\|_F^2$ is the squared Frobenius norm. Here, H_s^k and H_p^k are matrices where rows are the hidden vectors (see Eq. (1)) generated by the shared and private encoders given an input sentence of task k. The final loss function is a weighted combination of three parts:

$$L = L_{multi-task} + \beta L_{adv} + \gamma L_{diff}$$

where β and γ are hyper-parameters, $L_{multi-task}$ refers to a simple summation over the cross entropy loss for each task. We tune β and γ in the range $[0.001, 0.005, 0.01, 0.05, 0.1, 0.5]$ and present the best β and γ values in table 4 (provided in the appendix) for different multi-task learning settings.

3.2 Unifying Sentence Embeddings and Contextualized Vectors

Existing studies (Subramanian et al., 2018; Peters et al., 2018) leverage large amount of data to train sentence or word representations. However, sometimes it is impractical to assume there is an access to these large-scale data or computation resources. In these circumstances, we can combine existing encoders in a post-processing step to leverage the humongous data sources. In this work, we show that by combining our MTL based sentence encoder with an existing sentence encoder (Subramanian et al., 2018) and a contextualized word representation (Peters et al., 2018) encoder, we achieve state-of-the-art transfer performance on a wide variety of text classification tasks.

The contextualized word vectors refer to the hidden states generated by a BiLSTM (as in Eq. (1)) given a sequence of words (sentences) as inputs. To form a fixed size sentence representation from the contextual word vectors, we apply average pooling[3]. Although Peters et al. (2018) suggested learning the weights of the contextual word vectors, we do not learn any additional weights because we consider scenarios when there is no training example available to learn such weights. To get a universal sentence representation, we concatenate the sentence embeddings provided by our MTL based encoders, an existing sentence encoder (Subramanian et al., 2018) trained with another set of tasks, and the fixed size vector constructed from contextual word vectors (Peters et al., 2018). We will investigate more effective ways to combine multiple representations in our future work.

4 Experiments

In this section, we first show that our proposed sentence encoder can achieve state-of-the-art transfer performances. Then we demonstrate that combining the multi-task trained sentence representations with other sentence and word vectors yield better universal sentence representations. To better understand our results, we provide a thorough ablation study and confirm our combined encoder can learn robust and transferable sentence representations.

4.1 Experimental Setup

We use three large-scale textual entailment and paraphrasing tasks to train sentence encoders with multi-task learning, and combine these with an existing sentence encoder and contextualized word embeddings to compose the final sentence representations. We test the generalizability of the sentence embeddings on fifteen transfer tasks. A detailed description of the source and transfer tasks are presented in table 5 in the appendix. In addition, we perform a quantitative analysis using ten probing tasks to show what linguistic information is captured by our proposed sentence encoders.

Source tasks. The first two source tasks are natural language inference (NLI) which determines whether a natural language hypothesis can be inferred from a natural language premise. We consider the SNLI (Bowman et al., 2015) and the Multi-Genre NLI (MNLI) (Williams et al., 2017) which consist of sentence pairs, manually labeled with one of the three categories: entailment, contradiction and neutral. Following Conneau et al. (2017), we also conduct experiments that combine SNLI and MNLI datasets, which is denoted as AllNLI. The second task is the Quora question paraphrase (QQP)[4] detection based on a dataset of 404k question pairs. We use the Quora dataset split as that in Wang et al. (2017). We present and discuss the source task performances in appendix A.

Transfer and probing tasks. We evaluate the sentence encoders on fifteen transfer and ten probing tasks using the SentEval toolkit (Conneau & Kiela, 2018). Among the transfer tasks, six are text classification tasks for sentiment analysis (MR, SST), question-type (TREC), product reviews (CR), subjectivity/objectivity (SUBJ) and opinion polarity (MPQA). Rest of the transfer tasks (SICK-E, SICK-R, MRPC, STSB, and STS12–16) are semantic relatedness and textual similary tasks. We test our sentence encoder on capturing linguistic (surface, syntactic, and semantic) information using the ten probing tasks suggested in (Conneau et al., 2018).

Hyper-parameter tuning. We carefully tune the parameters and report the testing performance with best parameters. We use SGD with an initial learning rate of 0.1 and a weight decay of 0.99. At each epoch, we divide the learning rate by 5 if the development accuracy decreases. We use mini-batches of size 128 and training is stopped when the learning rate goes below the threshold of 10^{-5}. For the task-specific classifier, we use a multi-layer perceptron with 1 hidden-layer of 512 hidden units. We consider the range $[256, 512, 1024, 2048]$ for the number of hidden units in BiLSTM and found 2048 results in best performance. We use 300 dimensional GloVe word vectors (Pennington et al., 2014) trained on 840 billions of tokens as fixed word embeddings.

4.2 Evaluation on Transfer Tasks

We benchmark the performances of our proposed encoders on fifteen transfer tasks in comparison with several baselines including unsupervised models, sentence encoders, contextualized word encoders and supervised models. Table 1 and 2 summarize the results. From the block 4 of table 1, we see our combined encoders achieve best performance on 5/8 transfer tasks, which demonstrates the efficacy of our proposed unified sentence encoder.

[3] We tried max pooling, but it consistently performed more inferior compared to average pooling.

[4] https://www.kaggle.com/quora/question-pairs-dataset

Model Type	MR	CR	SUBJ	MPQA	SST	TREC	SICK-E	MRPC
1.Unsupervised sentence representations learning								
1.1. FastSent (Hill et al., 2016)	70.8	78.4	88.7	80.6	-	76.8	-	72.2/80.3
1.2. SkipThought (Kiros et al., 2015)	76.5	80.1	93.6	87.1	82.0	92.2	82.3	73.0/82.0
1.3. USE (Transformer) (Cer et al., 2018)	81.4	87.4	93.9	87.0	85.4	92.5	-	-
1.4. Byte mLSTM (Radford et al., 2017)	**86.9**	**91.4**	94.6	88.5	-	-	-	75.0/82.8
2. Supervised sentence representations learning								
2.1. InferSent (Conneau et al., 2017)	81.6	85.9	92.4	90.4	85.3	87.0	85.6	75.5/82.2
2.2. GenSen (Subramanian et al., 2018)	82.7	87.6	94.1	91.1	83.5	93.0	87.8	**78.0/84.2**
2.3. Sent2vec (SP) **(this paper)**	81.7	86.3	93.7	90.8	86.2	89.4	86.0	75.2/82.2
2.4. Sent2vec (ASP) **(this paper)**	82.3	86.3	93.5	90.8	84.2	89.4	87.1	76.3/83.2
3. Contextualized word vectors								
3.1. CoVe (McCann et al., 2017)	75.3	77.2	89.1	88.3	80.3	85.6	78.7	70.3/81.6
3.2. ELMo (Peters et al., 2018)	81.2	84.2	95.0	90.2	87.2	92.8	81.4	76.0/82.3
4. Sentence representations combined with contextualized word vectors **(this paper)**								
4.1. Sent2Vec (ASP) + ELMo	84.5	87.7	95.6	**91.7**	**88.7**	93.8	86.3	77.0/84.2
4.2. GenSen + ELMo	85.0	88.0	95.8	91.5	85.9	**94.8**	87.0	77.4/83.5
4.3. Sent2vec (ASP) + GenSen + ELMo	85.9	88.3	**96.2**	91.6	87.7	94.6	**88.4**	77.2/84.4
5. Approaches trained from scratch on the tasks								
5.1. AdaSent (Zhao et al., 2015)	83.1	86.3	95.5	93.3	-	92.4	-	-
5.2. TF-KLD (Ji & Eisenstein, 2013)	-	-	-	-	-	-	-	80.4/85.9
5.3. Illinois LH (Lai & Hockenmaier, 2014)	-	-	-	-	-	-	84.5	-
5.4. BLSTM-2DCNN (Zhou et al., 2016)	82.3	-	94.0	-	89.5	96.1	-	-

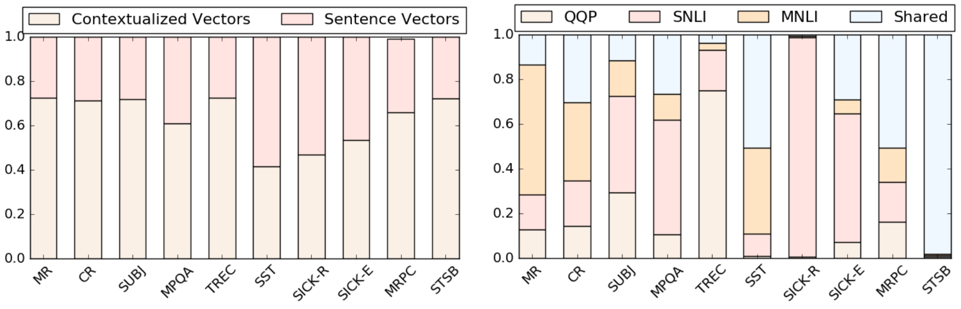
Table 1: Evaluation of sentence representations on a set of 8 tasks using a logistic regression classifier. “SP” and “ASP” in row 2.3 and 2.4 refers to the shared-private and adversarial shared private multi-task learning models. Values indicate the accuracy (accuracy/F1 for MRPC) for the test sets and bold-faced values denote the best *transfer* performances. We employ an averaging bag-of-words technique to form sentence embeddings, using features from all three layers of ELMo.

Model Type	SICK-R	STSB	Semantic Textual Similarity (STS)				
			2012	2013	2014	2015	2016
InferSent (Conneau et al., 2017)	0.884	0.756	**0.61**	0.56	0.68	0.71	0.71
GenSen (Subramanian et al., 2018)	0.888	0.786	**0.61**	0.54	0.65	0.74	0.67
USE (Transformer) (Cer et al., 2018)	0.860	**0.814**	**0.61**	**0.64**	**0.71**	**0.74**	0.74
CoVe (McCann et al., 2017)	0.808	0.745	0.50	0.40	0.63	0.54	0.61
ELMo (Peters et al., 2018)	0.813	0.661	0.55	0.49	0.62	0.67	0.62
Sent2vec (SP) **(this paper)**	0.887	0.752	0.55	0.56	0.64	0.67	0.72
Sent2vec (ASP) **(this paper)**	0.888	0.738	0.57	0.59	0.67	0.71	0.74
Sent2vec + ELMo **(this paper)**	0.888	0.720	0.60	0.55	0.66	0.70	0.70
Sent2vec + GenSen + ELMo **(this paper)**	**0.895**	0.753	**0.61**	0.55	0.67	0.71	**0.75**

Table 2: Transfer evaluation of the semantic relatedness and textual similarity tasks. “SP” and “ASP” in block 3 refers to the shared-private and adversarial shared private multi-task learning models. In block 4, we use ASP setting for Sent2vec. We use features from the top layer of the ELMo to produce sentence embeddings. Values indicate the Pearson correlation coefficient for the test sets and bold-faced values denote the best performance across all the models.

We further analyze the performance of our proposed combined encoders from two aspects: the improvement achieved by multi-task learned sentence encoders and the efficiency of the combination. From block 2 of table 1, we see the MTL based sentence encoders, Sent2vec outperforms the single task based sentence encoder InferSent on 7 out of 8 transfer tasks (comparing row 2.1 with 2.3–2.4). Sent2Vec also provides competitive performance on the other 7 tasks as shown in table 2. The results demonstrate that learning from multiple tasks helps to capture more generalizable features that are suitable for transfer learning. In addition, when using adversarial training, we observe improvements in 9 out of 15 transfer tasks comparing to the non-adversarial setting (see table 1 and 2). To investigate the advantages of adversarial training, we provide a detailed comparison between the shared and private encoders with and without adversarial training in appendix C.

Combining sentence encoders and contextualized word vectors improves the transfer learning performance significantly (comparing row 4.3 to 2.2, 2.4, and 3.2 in table 1). First, we see from table 2, when there are no training examples available for tasks (STS12–16 and STSB tasks), sentence embeddings (blocks 1 and 3) perform better than contextualized vectors (block 2). The result demon-



(a) (b)

Figure 1: (a) Weights learned by transfer tasks for contextualized vectors (ELMo, CoVe), and sentence vectors which refer to a concatenation of all the private and shared encoders of the adversarial shared-private multi-task model. (b) Weights learned by the transfer tasks for private (task-specific) and shared encoders of adversarial shared private multi-task model.

strates the necessity of learning generic sentence representations so that they can be directly used (without training) in transfer tasks. Although Sent2vec outperforms ELMo on 12/15 tasks (comparing row 2.4 to 3.2 in table 1 and row ELMo to Sent2vec (ASP) in table 2), it fell short to GenSen on most of the transfer tasks since GenSen is trained using 124M sentence pairs while Sent2vec is trained on 1.4M pairs. Because training an encoder on massive datasets require large computational resources, the direct utilization of GenSen through a post-processing step can bring benefits from large resources in a computational efficient way. Second, contextualized vectors (ELMo) perform better in specific tasks like SST. Also contextualized vectors are shown capable to capture specific linguistic properties (more details in the analysis part). These indicate that utilizing contextualized vectors may help learn better sentence representations. As a result, in the block 4 of table 1, when Sent2vec, GenSen, and ELMo are combined, we observe a significant improvement on 4 out of 8 tasks (MR, CR, SST, and SICK-E) over its individual components and competitive performance on the other tasks, which confirms the efficiency of the combination.

4.3 Analysis

Impact of sentence embeddings and contextualized word vectors. To analyze the contributions of our proposed sentence encoder and the sentence representations learned from contextualized word vectors (CoVe, ELMo) in a combined encoder during transfer learning, we design a classifier with a different network architecture. The classifier first generates predicted class probabilities based on a softmax layer using each sentence representations as input. Then the predictions are combined by a pooling layer with a weight parameter for each encoder. By investigating the learned weights in the pooling layer, we can understand which encoder contributes the most. The learned weights are shown in Figure 1(a). Although contextualized vectors have higher weights in 7/10 tasks, sentence vectors have $> 20\%$ contributions for each task and play dominant roles in tasks like SST and SICK-R. As we have shown in table 1, the combined encoder performs better than individual encoders (comparing row 4.1 to 2.4, 3.1, 3.2), indicating the contribution of the sentence and contextual word encoders are quite complementary.

Impact of source tasks on transfer tasks. To understand the influences of the source tasks on the transfer tasks, We conduct a similar analysis as in Figure 1(a) and show the learned weights assigned for the private (task-specific) and shared (generic) encoders in the ASP model in Figure 1(b). In general, for target tasks that are similar to the source tasks, the private encoders get higher weights, otherwise, the shared encoder is better. The combination of the shared and private encoders enables the transfer task to choose the best combination, thus achieved the best results. Most of the transfer tasks assign large weights on the SNLI task-specific and low weights on the QQP task-specific encoder, which explains why representations learned on SNLI are especially efficient for transfer tasks as noted in (Conneau et al., 2017). Besides, with adversarial training enforced, the shared encoder gets a lower weight from most of the transfer tasks than non-adversarial training (see figure 3 in the appendix) demonstrating the efficiency of adversarial training to separate generic and task-specific representations.

Model Type	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
InferSent	84.0	90.5	38.6	47.3	62.3	87.1	85.9	81.5	59.8	68.5
GenSen	93.9	**96.7**	44.0	64.1	75.5	90.0	90.1	90.3	53.6	69.0
USE (Transformer)	79.8	54.2	30.5	**68.7**	60.5	86.2	77.8	74.6	58.5	58.2
CoVe	88.4	19.0	41.2	39.1	68.8	86.2	84.6	85.4	50.8	61.3
ELMo	**95.3**	76.0	42.6	50.4	85.1	89.6	91.4	89.1	59.6	67.8
Sent2Vec (SP)	87.3	88.1	41.9	51.9	62.8	88.1	87.7	83.9	**60.6**	71.1
Sent2Vec (ASP)	88.0	85.5	41.0	53.2	53.1	88.3	87.3	83.3	49.9	70.8
Sent2Vec (ASP) + ELMo + GenSen	91.0	86.2	**45.2**	59.2	**85.4**	**91.0**	**92.4**	**91.7**	49.9	**73.3**

Table 3: Probing task accuracies with MLP as the classifier. For ELMo, the same bag-of-words averaging technique is employed as used for the downstream transfer tasks. When ELMo is combined with Sent2Vec and GenSen, features only from the top layer are used to fit in single GPU (Titan X). Bold-faced values denote the best results across the board.

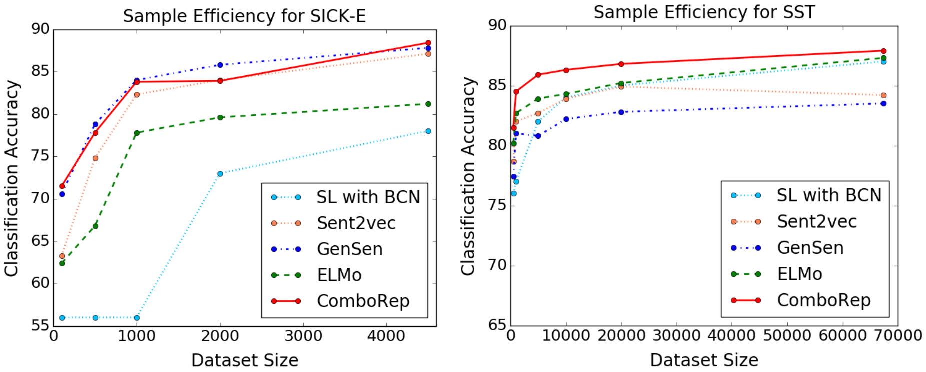


Figure 2: Comparing test performances of supervised learning (using BCN), word (ELMo) and sentence level representations (Sent2vec, GenSen) and their combination (ComboRep refers to Sent2vec + GenSen + ELMo) on SST and SICK-E tasks as the training dataset size is varied.

Probing for linguistic properties. To understand what linguistic properties are captured by our proposed sentence encoders and the unified encoders, we conduct experiments on probing tasks proposed in Conneau et al. (2018). Results are provided in table 3. Sentence representations show superiority in hard semantic tasks like SOMO and CoordInv, while contextualized word vectors perform better on embedding surface and syntactic properties. Moreover, MTL encoders (Sent2Vec) outperform both contextualized word vectors (ELMo) and the sentence encoder trained on single task (Infersent) on hard semantic tasks. The unified sentence encoder that combines both the sentence and contextual word representations captures most of the linguistic properties.

Impact of training data. Finally, we study the sample efficiency of the sentence and contextualized word encoders, as well as a strong supervised learning baseline, BCN (McCann et al., 2017), training from scratch on SST and SICK-E tasks. The results are shown in Figure 2. We see that the transfer setting have better sample efficiency especially when the training data is limited ($< 5,000$ samples). Besides, our proposed sentence encoder Sent2vec outperforms the GenSen encoder on the SST task but it fall short on the SICK-E task. We show that the combined sentence encoder has higher sample efficiency (can be trained with less labeled examples) than individual ones. We compare single and multi-task sentence encoders by varying the dataset size and present the results in the appendix B.

5 Conclusion

In this paper, we propose to leverage available large-scale text classification datasets and existing word and sentence encoding models to learn a universal sentence encoder. We utilize multi-task learning (MTL) to train sentence encoders that learn both generic and task-specific sentence representations from three heterogeneous text classification corpora. Experiments show that the MTL trained representations outperforms sentence encoders trained on singe task on a variety of transfer sentence classification tasks. We then further combine these sentence encoders with an existing multi-task pre-trained sentence encoder (with a different set of tasks) and a contextualized word representation learner. Our proposed unified sentence encoder yields significant improvements over the state-of-the-art sentence representations on transfer learning tasks. Extensive comparisons and thorough analysis using 15 transfer datasets and 10 linguistic probing tasks endorse the robustness of our proposed universal sentence encoder.

REFERENCES

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. VQA: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2425–2433, 2015.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. *International Conference on Learning Representations*, 2017.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Katherine Bailey and Sunny Chopra. Few-shot text classification with pre-trained word embeddings and a human in the loop. *arXiv preprint arXiv:1804.02063*, 2018.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

Colin Cherry and Hongyu Guo. The unreasonable effectiveness of word representations for twitter named entity recognition. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 735–745, 2015.

Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 conference on Empirical Methods in Natural Language Processing*, 2017.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255. IEEE, 2009.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and whats next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 473–483, 2017.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

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

Yangfeng Ji and Jacob Eisenstein. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 891–896, 2013.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pp. 3294–3302, 2015.

Alice Lai and Julia Hockenmaier. Illinois-lh: A denotational and distributional approach to semantics. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 329–334, 2014.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1188–1196, 2014.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *International Conference on Learning Representations*, 2017.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Deep multi-task learning with shared memory. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016a.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2016b.

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

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. Learning natural language inference using bidirectional lstm model and inner-attention. *arXiv preprint arXiv:1605.09090*, 2016c.

Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. *International Conference on Learning Representations*, 2018.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. In *International Conference on Learning Representations*, 2016.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pp. 6297–6308, 2017.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. How transferable are neural networks in nlp applications? *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016.

Hao Peng, Sam Thomson, and Noah A Smith. Deep multitask learning for semantic dependency parsing. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

Nanyun Peng and Mark Dredze. Multi-task domain adaptation for sequence tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 91–100, 2017.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*, pp. 1532–1543, 2014.

Christian S Perone, Roberto Silveira, and Thomas S Paula. Evaluation of sentence embeddings in downstream and linguistic probing tasks. *arXiv preprint arXiv:1806.06259*, 2018.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*, 2017.

Marek Rei. Semi-supervised multitask learning for sequence labeling. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *International Conference on Learning Representations*, 2017.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 806–813, 2014.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. In *International Conference on Learning Representations*, 2018.

Damien Teney, Lingqiao Liu, and Anton van den Hengel. Graph-structured representations for visual question answering. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Subhashini Venugopalan, Lisa Anne Hendricks, Marcus Rohrbach, Raymond Mooney, Trevor Darrell, and Kate Saenko. Captioning images with diverse objects. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Shuohang Wang and Jing Jiang. Learning natural language inference with lstm. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral multi-perspective matching for natural language sentences. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. In *International Conference on Learning Representations*, 2016.

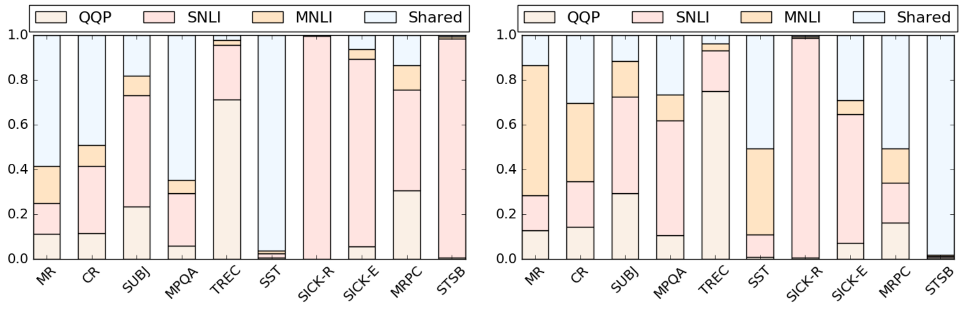
Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. Multi-task label embedding for text classification. *arXiv preprint arXiv:1710.07210*, 2017a.

Honglun Zhang, Liqiang Xiao, Yongkun Wang, and Yaohui Jin. A generalized recurrent neural architecture for text classification with multi-task learning. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017b.

Han Zhao, Zhengdong Lu, and Pascal Poupart. Self-adaptive hierarchical sentence model. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pp. 4069–4076, 2015.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. *arXiv preprint arXiv:1611.06639*, 2016.



(a)

(b)

Figure 3: Weights learned by the transfer tasks for private (task-specific) and shared encoders of shared private multi-task model (a) with and (b) without adversarial training.

Tasks	β	γ
QQP and SNLI	0.01	0.05
SNLI and MNLI	0.005	0.001
QQP and AllNLI	0.01	0.05
QQP, SNLI and MNLI	0.005	0.001

Table 4: Best β and γ values for adversarial shared private model on different set of tasks.

Name	N	V	Task	C
Binary and multi-class classification tasks				
MR	11k	20.3k	sentiment	2
CR	4k	5.7k	product review	2
SUBJ	10k	22.6k	subj/obj	2
MPQA	11k	6.2k	opinion	2
SST	70k	17.5k	sentiment	2
TREC	6k	9.7k	question-type	6
Recognizing textual entailment tasks				
SNLI[†]	560k	42.7k	entailment	3
MNLI[†]	433k	102.7k	entailment	3
SICK-E	10k	2.4k	entailment	3
Paraphrase identification tasks				
QQP[†]	404k	127.5k	paraphrasing	2
MRPC	5.8k	19.5k	paraphrasing	2
Semantic textual similarity tasks				
SICK-R	10k	2.4k	similarity	0 – 5
STSB	8.6k	15.9k	similarity	5
STS-12	399	735	similarity	0 – 5
STS-13	561	1.6k	similarity	0 – 5
STS-14	750	3.8k	similarity	0 – 5
STS-15	750	1.3k	similarity	0 – 5
STS-16	209	868	similarity	0 – 5

Table 5: Statistics of the datasets for multi-task learning and the transfer tasks. N is the number of samples, V is the vocabulary size, and C is the number of classes or score range. [†] denotes the datasets that are used in multi-task learning.

A Evaluation on Source Tasks

In this section, we discuss the performance of the shared-private multi-task learning (MTL) frameworks on different combinations of QQP, SNLI and MNLI datasets as the *source tasks*. We concatenate the representations generated by the shared and private encoders to form sentence embeddings. The results are presented in table 6. We compare the performance of MTL with the models trained on single tasks as in (Conneau et al., 2017). Table 6 shows that learning from multiple tasks performs better than learning from a single task. However, to our surprise, the adversarial training does not *always* excel on source tasks but we show in the transfer evaluation that adversarial training boosts the transfer learning performance.

Model Type	QQP		SNLI		MNLI	
	dev	test	dev	test	dev	test
Learning from in-domain single task						
(Conneau et al., 2017)	87.1	86.7	84.7	84.5	70.2/70.8	70.8/69.8
Learning from 2-datasets and 2-tasks (SNLI and MNLI)						
Shared-Private	-	-	85.0	**85.3**	**71.7/71.4**	**71.8/70.6**
Adversarial Shared-Private	-	-	84.9	84.9	70.9/71.4	71.0/70.0
Learning from 2-datasets and 2-tasks (QQP and SNLI)						
Shared-Private	87.0	86.8	84.8	84.7	-	-
Adversarial Shared-Private	87.5	**87.0**	**85.2**	84.9	-	-
Learning from 3-datasets and 2-tasks (QQP and AllNLI)						
Shared-Private	86.9	86.0	**85.2**	84.7	70.7/70.5	70.8/69.3
Adversarial Shared-Private	87.0	86.6	84.7	84.3	70.2/69.4	69.6/68.3
Learning from 3-datasets and 3-tasks (QQP, SNLI and MNLI)						
Shared-Private	**87.6**	87.0	**85.2**	85.2	71.2/71.0	71.0/70.1
Adversarial Shared-Private	86.6	86.3	84.6	84.7	70.7/70.7	71.0/70.1

Table 6: Validation and test accuracy of the source tasks obtained through various multi-task learning architectures. Bold-faced values indicate best performance across all the models.

Model Type	MR	CR	SUBJ	MPQA	SST	TREC	SICK-R	SICK-E	MRPC	STS14
Sentence representation learning from single-task										
BiLSTM-Max (on SNLI)	80.1	85.3	92.6	89.1	83.6	89.2	0.885	86.0	75.2/82.4	.66/.64
BiLSTM-Max (on QQP)	79.2	84.6	92.6	88.8	83.5	88.0	0.861	82.4	74.8/82.8	.62/.60
BiLSTM-Max (on MNLI)	81.2	85.8	93.1	89.5	83.4	88.8	0.863	84.7	75.9/83.1	.66/.63
BiLSTM-Max (on AllNLI)	80.9	86.3	93.2	89.2	83.3	88.8	0.887	86.7	76.4/83.4	.69/.66
Sentence representation learning from two-tasks (QQP and SNLI)										
Shared-Private	80.5	84.8	93.4	89.1	84.0	90.2	0.881	86.1	75.1/83.2	.65/.62
Adversarial Shared-Private	80.9	85.4	93.4	89.2	83.6	90.8	0.886	86.9	76.5/82.9	.68/.65
Sentence representation learning from two-tasks (SNLI and MNLI)										
Shared-Private	81.7	86.4	93.7	89.6	84.8	89.2	0.885	86.7	76.3/82.7	.67/.64
Adversarial Shared-Private	81.2	86.0	93.0	89.3	83.7	90.4	0.886	**87.1**	76.9/83.5	.70/.67
Sentence representation learning from two-tasks (QQP and AllNLI)										
Shared-Private	**82.0**	86.1	**93.9**	89.4	84.6	89.6	0.884	86.3	76.4/83.4	.68/.64
Adversarial Shared-Private	81.4	86.3	93.2	89.4	85.1	88.4	**0.888**	86.6	75.5/82.5	.67/.63
Sentence representation learning from three-tasks (QQP, SNLI and MNLI)										
Shared-Private	81.6	86.9	**93.9**	89.2	84.4	90.4	0.883	85.9	76.5/83.3	.66/.63
Adversarial Shared-Private	**82.0**	86.3	93.8	89.4	84.1	**92.2**	0.884	87.0	**77.2/83.6**	.68/.65

Table 7: Transfer test results for various single-task and multi-task learning architectures trained on a combination of QQP, SNLI and MNLI datasets. Bold-faced values indicate the best performance among all models in this table.

Data Size	MR	CR	SUBJ	MPQA	SST	TREC	SICK-R	SICK-E	STS14	MRPC
Same for MTL, STL	+0.1	+0.6	+1.6	−0.7	+0.0	+1.7	−0.008	−1.5	+0.7	−0.003
Larger for MTL	+1.0	+0.8	+1.3	−0.6	+0.3	+2.1	+0.001	+0.6	+1.4	+0.0

Table 8: The accuracy differences between MTL and STL when training with different sizes of data. For the same data size, MTL and STL are trained on equal amount of annotated data. For larger data size for MTL, MTL is trained on two datasets while STL on one dataset (less data).

B Singel-task vs. multi-task learning with varying training data

When we compare MTL to STL for transfer learning, one fundamental question that arises is, does improvement in transfer learning via MTL only come because of having more annotated data? Comparing the performance of AllNLI in a single task setting and {SNLI, MNLI} in the multi-task settings in table 7, we observe significant improvement in 7/10 tasks. In both settings, the amount of training data is the same. To verify the hypothesis that the improvements in transfer learning do not solely come from having more annotated data, we design an experiment that samples equal amount of data (225k training examples) from SNLI and QQP to match the size of full SNLI dataset. We found 0.26% average improvement in transfer tasks compared to single task learning (STL) on the SNLI dataset. With full SNLI and QQP dataset, we observe a larger (0.69% on average) improvement in transfer tasks compared to STL on SNLI dataset. The first row of table 8 shows that MTL is beneficial in this setting and the second row demonstrates that with additional data, MTL achieves larger improvements.

C Private Encoders vs. Shared Encoder

To verify our hypothesis that shared encoder learns generic features that are more suitable for transfer learning and with adversarial training enforced, the shared encoder becomes more effective; we provide a detailed comparison of the private and shared encoders in table 9. However, by concatenating the shared and task-specific representations, we can achieve better transfer performance which indicates that transfer tasks also get benefited from task-specific features, specially when the source and transfer tasks are homogeneous (more details are provided in the ablation analysis).

Model Type	MR	CR	SUBJ	MPQA	SST	TREC	SICK-R	SICK-E	MRPC	STS14
Shared-Private (trained on SNLI and MNLI)										
Private Encoder (on SNLI)	79.5	84.0	92.7	89.1	82.0	87.8	0.881	84.8	75.0/82.7	.65/.63
Private Encoder (on MNLI)	80.6	84.6	92.7	89.2	82.9	88.0	0.853	83.8	75.1/82.9	.60/.58
Shared Encoder	80.9	86.4	92.9	89.5	84.0	88.0	0.879	84.8	75.8/83.1	.69/.65
Combined Encoder	81.7	86.4	93.7	**89.6**	84.8	89.2	0.885	86.7	76.3/82.7	.67/.64
Adversarial Shared-Private (trained on SNLI and MNLI)										
Private Encoder (on SNLI)	79.4	84.6	92.1	89.0	82.8	86.6	0.886	85.5	74.0/81.4	.68/.66
Private Encoder (on MNLI)	80.4	84.6	92.6	89.1	83.3	86.8	0.863	83.4	76.0/83.3	.65/.63
Shared Encoder	81.2	86.7	92.4	89.3	84.5	87.0	0.875	85.1	74.8/82.6	.56/.57
Combined Encoder	81.7	86.5	93.4	89.5	**84.9**	90.0	**0.888**	**87.1**	76.4/83.4	.64/.63
Shared-Private (trained on QQP and AllNLI)										
Private Encoder (on QQP)	79.4	82.6	92.5	88.5	82.2	89.2	0.856	82.9	74.3/82.4	.63/.59
Private Encoder (on AllNLI)	81.1	86.1	92.9	89.5	83.9	90.2	0.876	85.0	76.0/83.3	.67/.64
Shared Encoder	81.2	85.6	93.1	89.2	83.4	88.2	0.880	85.3	75.7/82.8	.69/.66
Combined Encoder	**82.0**	86.1	**93.9**	89.4	84.6	89.6	0.884	86.2	76.4/83.4	.68/.64
Adversarial Shared-Private (trained on QQP and AllNLI)										
Private Encoder (on QQP)	79.2	81.7	92.1	88.7	80.8	86.6	0.865	83.8	74.1/82.2	.67/.64
Private Encoder (on AllNLI)	81.5	86.5	92.8	89.4	82.9	88.4	0.885	85.5	75.7/83.2	**.70/.67**
Shared Encoder	80.5	84.8	92.6	89.2	83.2	82.6	0.876	84.8	75.5/83.1	.57/.56
Combined Encoder	81.9	85.9	93.0	**89.6**	82.4	90.6	0.887	86.7	76.8/83.3	.61/.60
Shared-Private (trained on QQP, SNLI and MNLI)										
Private Encoder (on QQP)	78.9	83.5	91.7	88.3	81.4	88.6	0.850	82.1	72.6/81.1	.62/.59
Private Encoder (on SNLI)	79.1	83.9	92.7	89.0	81.3	88.4	0.880	85.2	74.0/82.0	.66/.63
Private Encoder (on MNLI)	81.0	85.7	93.1	89.3	82.9	89.2	0.850	84.0	74.5/82.9	.60/.58
Shared Encoder	80.9	85.8	92.9	89.2	82.7	85.2	0.878	85.8	76.0/83.1	.68/.65
Combined Encoder	81.6	**86.9**	**93.9**	89.2	84.4	90.4	0.883	85.9	76.5/83.2	.66/.63
Adversarial Shared-Private (trained on QQP, SNLI and MNLI)										
Private Encoder (on QQP)	78.9	82.3	92.2	88.8	82.3	87.2	0.855	83.0	74.3/82.2	.64/.62
Private Encoder (on SNLI)	79.6	84.4	92.0	89.0	82.7	88.2	0.881	85.4	74.6/82.4	.67/.65
Private Encoder (on MNLI)	80.6	84.7	93.0	89.1	83.6	89.2	0.863	84.8	75.8/82.8	.65/.62
Shared Encoder	81.0	85.9	92.7	**89.6**	82.9	87.0	0.876	85.7	74.8/82.8	.66/.64
Combined Encoder	**82.0**	86.3	93.8	89.4	84.1	**92.2**	0.884	87.0	**77.2/83.6**	.66/.64

Table 9: Detailed analysis of the transfer test results for shared-private models trained on different combinations of QQP, SNLI and MNLI datasets. Combined encoder refers to the concatenation of shared encoder and all private encoders. Underlined values indicate the best performance among different encoders of the shared-private models trained on the same set of tasks. Bold-faced values indicate the best performance among all models in this table.