Dropping Networks for Transfer Learning

Anonymous CoNLL submission

Abstract

In natural language understanding, many challenges require learning relationships between two sequences for various tasks such as similarity, relatedness, paraphrasing and question matching. Knowledge transfer can be quite effective for closely related tasks. However, transferring all knowledge, some of which irrelevant for a target task, can lead to sub-optimal results due to *negative* transfer. Hence, this paper focuses on the transferability of both instances and parameters across natural language understanding tasks using an ensemble-based transfer learning method to circumvent such issues. The primary contribution of this paper is the combination of both *Dropout* and *Bagging* for improved transferability in neural networks, referred to as *Dropping* herein. Secondly, we present a straightforward yet novel approach for incorporating source *Dropping* Networks to a target task for few-shot learning that mitigates *negative* transfer. This is achieved by using a decaying parameter chosen according to the slope changes of a smoothed spline error curve at sub-intervals during training. We compare the proposed approach against hard parameter sharing and soft parameter sharing transfer methods in the few-shot learning case. We also compare against models that are fully trained on the target task in the standard supervised learning setup. The aforementioned adjustment leads to improved transfer learning performance and comparable results to the current state of the art only using a fraction of the data from the target task.

1 Introduction

Learning relationships between sentences is a fundamental task in natural language understanding (NLU). Given that there is gradience between words alone, the task of scoring or categorizing two sentences is made even more challenging, particularly when either sentence are less grounded and more conceptually abstract. This research area has been active since the study of compositional semantics that represented hierarchical compositions in logical form (Mitchell and Lapata, 2010). Since then, distributed representations in the form of word or sub-word vectors have dramatically improved models when coupled with neural networks for supervised learning, as seen for distributed compositional semantic models e.g pairwise-based neural networks for textual entailment, paraphrasing and relatedness scoring (Mueller and Thyagarajan, 2016).

Therefore, we focus on such tasks to evaluate our proposed transfer learning approach. Hence, we start by providing a brief description of the datasets used. We show the model averaging properties of *Dropping* networks show significant benefits over *Bagging* neural networks or a single neural network with *Dropout*, particularly when dropout is high (p=0.5), leading to greater diversity and specialization in each model. Additionally, we find that distant tasks that have some knowledge transfer can be overlooked if possible effects of negative transfer are not addressed. The proposed weighting scheme takes this issue into account, improving over alternative approaches as we will see in Section 5.

2 Dataset Description

2.1 Natural Language Inference

Natural Language Inference (NLI) deals with inferring whether a hypothesis is true given a premise. Such examples are seen in entailment and contradiction. The Stanford Natural Language Inference (SNLI) dataset introduced by Bowman et al. (2015) provides the first large scale corpus with a total of 570K annotated sentence pairs (much larger than previous semantic matching datasets such as the *SICK* (Marelli et al., 2014) dataset that consisted

of 9927 sentence pairs). As described in the opening statement of McCartney's thesis (MacCartney, 2009), "the emphasis is on informal reasoning, lexical semantic knowledge, and variability of linguistic expression." The SNLI corpus addresses issues with previous manual and semi-automatically annotated datasets of its kind which suffer quality, scale and entity co-referencing that leads to ambiguous and ill-defined labeling. They do this by grounding the instances with a given scenario which leaves a precedent for comparing the contradiction, entailment and neutrality between premise and hypothesis sentences.

Since the introduction of this large annotated corpus, further resources for multi-genre NLI (MultiNLI) have recently been made available as apart of a Shared RepEval task (Nangia et al., 2017; Williams et al., 2017). MultiNLI extends a 433k instance dataset to provide a wider coverage containing 10 distinct genres of both written and spoken English, leading to a more detailed analysis of where machine learning models perform well or not, unlike the original SNLI corpus that only relies only on image captions. As authors describe, "temporal reasoning, belief, and modality become irrelevant to task performance" are not addressed by the original SNLI corpus. Another motivation for curating the dataset is particularly relevant to this problem, that is the evaluation of transfer learning (TL) across domains, hence the inclusion of these datasets in the analysis. These two NLI datasets allow us to analyze the transferability for two closely related datasets.

2.2 Question Matching

Question matching (QM) is a relatively new pairwise learning task in NLU for semantic relatedness, the increased interest can be greatly attributed to the release of the Quora dataset provided in the form of a Kaggle competition[1]. The task has implications for Question-Answering (QA) systems and more generally, machine comprehension. A known difficulty in QA is the problem of responding to a question with the most relevant answers. In order to respond appropriately, grouping and relating similar questions can greatly reduce the possible set of correct answers that a neural network predicts. The Quora team have developed models for this task (quo), starting with hand crafted features, and then further developing a SN that combines encodings in a dense network upstream. The Quora challenge has a few characteristics that are worth pointing out from the onset. The dataset is created so that a unique question, or questions identical in intent, are not paired more than once. This ensures that a classifier does not require many pairings of the same questions to learn as in practice the likelihood of the exact same question being asked is relatively low. However, questions can appear in more than one instance pair $(\mathcal{S}_i^1, \mathcal{S}_i^2)$. In this work we ensure duplicates are not tested upon if at least one pair of the duplicate is used for single-task learning. Secondly, it is worth noting there has been speculation on the sampling used to generate the question pairs, and if so, does it skew the results by introducing bias, as the sampling method could be bias towards various topics in Quora, this is worth noting for the subsequent supervised TL.

[1] see here: https://www.kaggle.com/c/quora-question-pairs

3 Related Work

3.1 Neural Network Transfer Learning

In TL we seek to transfer knowledge from a one or more source task $\mathcal{T}_s$ in the form of instances, parameters and/or external resources to improve performance on a target task $\mathcal{T}_t$. This work is concerned about improving results in this manner, but also not to degrade the original performance of $\mathcal{T}_s$, referred to as *Sequential Learning*. In the past few decades, research on transfer learning in neural networks has predominantly been parameter based transfer. Yosinski et al. (2014) have found lower-level representations to be more transferable than upper-layer representations since they are more general and less specific to the task, hence negative transfer is less severe. This paper will later describe a method for overcoming this using an ensembling-based method, but before we note the most relevant work on transferability in neural networks.

Pratt et al. (1991) introduced the notion of parameter transfer in neural networks, also showing the benefits of transfer in structured tasks, where transfer is applied on an upstream task from its subtasks. Further to this (Pratt, 1993), a hyperplane utility measure as defined by θ_s from $\mathcal{T}_t$ which then rescales the weight magnitudes was shown to perform well, showing faster convergence when transferred to $\mathcal{T}_t$.

Raina et al. (Raina et al., 2006) focused on constructing a covariance matrix for informative Gaussian priors transferred from related tasks on binary text classification. The purpose was to over-

come poor generalization from weakly informative priors due to sparse text data for training. The off-diagonals of $\sum$ represent the parameter dependencies, therefore being able to infer word relationships to outputs even if a word is unseen on the test data since the relationship to observed words is known. More recently, TL in neural networks has been predominantly studied in Computer Vision (CV). Models such as AlexNet allow features to append to existing networks for further fine tuning on new tasks (Zheng et al., 2016). They quantify the degree of generalization each layer provides in transfer and also evaluate how multiple CNN weights are used to be of benefit in TL. This also reinforces to the motivation behind using ensembles in this paper.

3.1.1 Transferability in Natural Language

Mou et al. (2016) describe the transferability of parameters in neural networks for NLP tasks. Questions posed included the transferability between varying degrees of "similar" tasks, the transferability of different hidden layers, the effectiveness of hard parameter transfer and the use of multi-task learning as opposed to sequential based TL. They focus on transfer using hard parameter transfer, most relevantly, between SNLI and SICK. They too find that lower level features are more general, therefore more useful to transfer to other similar task, whereas the output layer is more task specific. Another important point raised in their paper was that a large learning rate can result in the transferred parameters being changed far from their original transferred state. As we will see, the method proposed here will inadvertently address this issue since the learning rates are kept intact within the ensembled models, a parameter adjustment is only made to their respective weight in a vote.

3.2 Dropout and Bagging Connection

Here we briefly describe past work that describe the connections between both Dropout (parameter-based) and Bagging (instance-based) model averaging techniques. Most notably, Baldi and Sadowski (Baldi and Sadowski, 2014, 2013) study the model averaging properties of dropout in neural networks with logistic and ReLU units, the dropout rate, dropping activation units and/or weights, convergence of dropout and the type of model averaging that is being achieved using dropout. They point out that dropout is performing SGD over the global ensemble error from subnetworks online instead of over the instances. Warde et al. (Warde-Farley et al., 2013) provide empirical results on the performance of dropout in ANN's that use ReLU activation functions and compare the geometric mean used in dropout to the arithmetic mean used in ensembles (such as Bagging). Gal and Ghahramani (Gal and Ghahramani, 2016) give a Bayesian perspective on dropout, casting dropout as approximate Bayesian inference in deep Gaussian Processes, interpreting dropout as a way to account for model uncertainty.

Concretely, dropout is a model averaging technique for ANN's that uses the geometric mean instead of arithmetic mean that is used for Bagging. In dropout the weights are shared in a single global model, whereas in ensembles the parameters are different for each model. Combining both is particularly suitable for avoiding negative transfer as models within the ensemble that perform well between $\mathcal{T}_s$ and $\mathcal{T}_t$ can be given a higher weight α than those that produce higher accuracy on $\mathcal{T}_t$. One strategy would be to solely rely on parameter transfer during training, considering only subnetworks induced via dropout, however, it is not clear when to decide the checkpoints that are most suitable to retrieve subnetwork weights that avoid negative transfer in particular. Hence, we rely on Bagging to somewhat mitigate this issue, yet still provide the generalization benefits that geomtric-based model averaging has shown to provide.

3.3 Pairwise Model Architectures

Before discussing results we describe the current SoTA for pairwise learning in Natural Language Understanding (NLU). Wang et al. (Wang et al., 2017a) recently proposed a self-matching attention network for reading comprehension style question answers that incorporates the use of pointer networks to identify the answer position from a given passage, showing top results on the Stanford Question Answering dataset (SQuAD) (Rajpurkar et al., 2016), using ensembling showed a 75.9%, a 4.6 percentage point increase over the single model. The similarity of this approach with ours is that a GRU network with standard soft attention is used to align the question and passage. Wang et. al (Wang et al., 2017b) also describe a Bilateral Multi-Perspective Matching model that proposes to overcome limitations in encoding of sentence representations by considering interdependent interactions between sentence pairs, likewise, we too offer a co-attention

mechanism between hidden layers to address this. Their work attempts this by first encoding both sentences separately with a BiLSTM and then match the two encoded sentences in two directions, at each timestep, a sentence is matched against all time-steps of the other sentence from multiple perspectives. Then, another BiLSTM layer is utilized to aggregate the matching results into a fixed-length matching vector, a prediction is then made through a fully connected layer. This was demonstrated on NLI, Answer Selection and Paraphrase Identification.

Shen et al. (2017) use a Word Embedding Correlation (WEC) model to score co-occurrence probabilities for Question-Answer sentence pairs on Yahoo! Answers dataset and Baidu Zhidao Q&A pairs using both a translation model and word embedding correlations. The objective of the paper was to find a correlation scoring function where a word vector is given while modelling word co-occurrence given as $C(q_i, \alpha_j) = (v_{q_i}^T / ||v_{q_i}||) \times (\mathbf{M}v_{a_j} / \mathbf{M}||v_{a_j}||)$, where M is a correlation matrix, v_q a word vector from a question and a word vector v_a from an answer. The scoring function was then expanded to sentences by taking the maximum correlated word in answer in a question divided by the answer length.

Parikh et al. (2016) present a decomposable attention model for soft alignments between all pairs of words, phrases and aggregations of both these local substructures. The model requires far less parameters compared to attention with LSTMs or GRUs. This paper uses attention in an SN by proposing attention across hidden layer representations of sentences, in an attempt to mimic how humans compare sentences. Weights are often tied in networks, according to the symmetric property $(\mathcal{S}_i^1, \mathcal{S}_i^2)$.

Yang et al. (2017) have described a character-based intra attention network for NLI on the SNLI corpus, showing an improvement over the 5-hidden layer BiLSTM network introduced by Nangia et al. (2017) used on the MultiNLI corpus. Here, the architecture also looks to solve to use attention to produce interactions to influence the sentence encoding pairs. Originally, this idea was introduced for pairwise learning by using three Attention-based Convolutional Neural Networks (Yin et al., 2015) that use attention at different hidden layers and not only on the word level. Although, this approach shows good results, word ordering is partially lost in the sentence encoded interdependent representations in CNNs, particularly when max or average pooling is applied on layers upstream.

Chen et al. (2017a) currently provide the best performing model on SNLI, by incorporating external knowledge from WordNet (Miller, 1995) and Freebase (Bollacker et al., 2008) in the co-attention mechanism. This accounts for local information between sentences, instead of encoding fixed representations of each sentence separately. They demonstrated that attention aided by external resources can improve the local interdependent interactions between sentences. They also use a knowledge-enriched inference collection which refers to comparing the normalized attention weight matrices both row-wise and column-wise to model local inference between word pair alignments "where a heuristic matching trick with difference and element-wise product " (Mou et al., 2015; Chen et al., 2017b) is used. In fact, in Mou et al.'s (Mou et al., 2015) work, they somewhat address the word ordering problem with a standard CNN for NLI by using a tree-based CNN that attempts to keep the compositional local order of words intact.

4 Methodology

This section describes data augmentation, pairwise learning, the use of attention and how the proposed TL method is used for initialization across the tasks. In fact, the more complex architectures become, the more difficult it becomes to study and interpret empirical results between *Dropping* transfer and its comparable hard parameter transfer baseline.

Data Augmentation For both Stanford NLI (SNLI) and Multi-NLI (MNLI) the class distribution is almost even therefore no re-weighting or sampling is required in these cases. However, due to the slight imbalance in the Quora dataset (36% matching questions and the remaining non-matching questions) a weighted Negative Log-Likelihood (NLL) loss function is used to account for the slight disproportion in classes. Another strategy is to upsample by reordering S_1 and S_2 to improve generalization, this is allowed because the semantics should be symmetric in comparison.

4.1 Baselines

For single-task learning, the baseline proposed for evaluating the co-attention model and the ensemble-based model consists of a standard GRU network with varying architecture settings for all three

datasets. During experiments we tested different combinations of hyperparameter settings. All models are trained for 30,000 epochs, using a dropout rate $p = 0.5$ with Adaptive Momentum (ADAM) gradient based optimization (Kingma and Ba, 2014) in a 2-hidden layer network with an initial learning rate $\eta = 0.001$ with a batch size $b_T = 128$. As a baseline for TL we use hard parameter transfer with fine tuning on 50% of $X \in \mathcal{T}_s$ of upper layers.

For comparison to other transfer approaches we note previous findings by Yosinski et al. (2014) which show that lower level features are more generalizable. Hence, it is common that lower level features are transferred and fixed for $\mathcal{T}_t$ while the upper layers are fine tuned for the task, as described in Section 3.3. Therefore, the baseline comparison simply transfers all weights from $\theta_s \rightarrow \theta_t$ from a global model instead of ensembles and these parameters as initialization before training on few examples on $\mathcal{T}_t$. Although, negative transfer can occur if the more generalizable lower level representations include redundant or irrelevant examples for the $\mathcal{T}_t$. Instead, here we are allowing the $\mathcal{T}_t$ to guide the lower level feature representations based on a weighted vote in the context of a decaying ensemble-based regularizer.

4.2 Attention

Encoded representations for paired sentences are obtained from $(\vec{h}_{T_1}^{(l+1)}, \vec{h}_{T_2}^{(l+1)})$ where $\vec{h}^{(l+1)}$ represents the last hidden layer representation in a neural network, for recurrent models this is obtained from at the last time step T. Since longer dependencies are difficult to encode and by only using the last hidden state as the context vector c_t, words at the beginning of a sentence have diminishing effect as the sentence becomes longer. One way of overcoming these sub-optimal sentence representations is to attend to the most salient words using the commonly used *Attention Mechanism*. Instead of using the final hidden layer representation for both (S_1, S_2), the output of hidden layers at each time step t is passed to an attention mechanism that acts as a weighting mechanism. The `softmax` function produces the attention weights α by passing all outputs of the source RNN, h_S to the softmax conditioned on the target word of the opposite sentence h_t. A context vector c_t is computed as the sum of the attention weighted outputs by $\bar{h}_s$. This results in a matrix $A \in \mathbb{R}^{|S| \times |T|}$ where $|S|$ and $|T|$ are the respective sentence lengths, in this work, the max

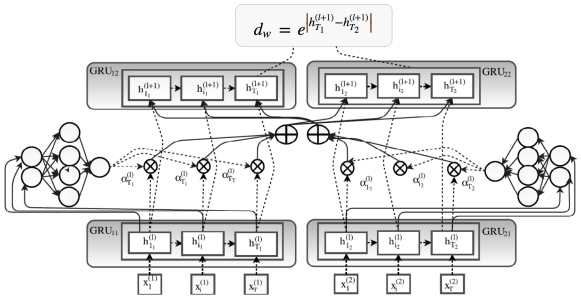

Figure 1: *Cross-Attention GRU-Siamese Network*

length of a given batch. The final attention vector α_t is used as a weighted input of the context vector c_t and the hidden state output h_t parameterized by a xavier uniform intialized weight vector W_c to a hyperbolic tangent unit.

4.3 Learning to Transfer

Here we note two methods that are considered for accelerating learning on $\mathcal{T}_t$ given the parameters of a learned model from $\mathcal{T}_s$. We first start by describing a method that learns to guide weights on $\mathcal{T}_t$ by measuring similarity between $\theta_{\hat{s}}$ and $\theta_{\hat{t}}$ during training by using moving averages on the slope of the error curve, followed by a description on the use of smoothing splines to avoid large changes due to volatility in the error curve during training.

***Dropping* Transfer** Both dropout and bagging are common approaches for regularizing models, the former is commonly used in neural networks. Dropout trains a number of subnetworks by dropping parameters and/or input features during training while also have less parameter updates per epoch. Bagging trains multiple models by sampling instances $\vec{x}_k \in \mathbb{R}^d$ from a distribution $p(\vec{x})$ (e.g uniform distribution) prior to training. Herein, we refer to using both in conjunction as *Dropping*.

Dropping Networks are similar to Adaptive Boosting (AdaBoost) in that there is a weight assigned based on performance during training. However, Dropping Networks weights are assigned based on the performance of each batch after Bagging, instead of each data sample. Furthermore, the use of Dropout promotes sparsity, combining both arithmetic mean and geometric mean model averaging. Avoiding negative transfer with standard AdaBoost is too costly in practice too use on large datasets and is prone to overfitting in the presence of noise (Mason et al., 2000).

A fundamental concern in TL is that we do not

want to transfer irrelevant knowledge which leads to slower convergence and/or sub-optimal performance. Therefore, dropping allows to place soft attention based on the performance of each model from $\mathcal{T}_s \rightarrow \mathcal{T}_t$ using a `softmax` as a weighted vote. Once a target model f_t is learned from only few examples on $\mathcal{T}_t$ (referred to as few-shot learning), the weighted ensembled models from $\mathcal{T}_s$ can be transferred and merged with the $\mathcal{T}_t$ model. Equation 1 shows the simple weighted vote between models where N is the number of ensembled models each of which have batch size S, ϕ denotes the `softmax` function and $\bar{a}_s^l$ denotes weighted average output from the ensembles trained on subsets of $\mathcal{T}_s$.

$$\bar{a}_s^l = \sum_{i=1}^{N} \alpha_i \Big(\frac{1}{S} \sum_{s=1}^{S} \phi(z_{s_i}^l) \Big) \quad s.t, \quad \sum_{i=1}^{N} \alpha_i = 1 \tag{1}$$

Equation 2 then shows a straightforward update rule that decays the importance of $\mathcal{T}_s$ *Dropping* networks as the $\mathcal{T}_t$ neural network begins to learn from only few examples. The prediction from few samples a_t^l is the single output from $\mathcal{T}_t^l$ and γ is the slope of the error curve that is updated at regular intervals during training.

We expect this approach to lead to faster convergence and more general features as the regularization is in the form of a decaying constraint from a related task. The rate of the shift towards the $\mathcal{T}_t$ model is proportional to the gradient of the error $\nabla_{x_{\tilde{s}}}$ for a set of mini-batches $x_{\tilde{s}}$. In our experiments, we have set the update of the slope to occur every 100 iterations.

$$\hat{y}_t = \gamma \bar{a}_s^l + (1-\gamma) a_t^l \quad s.t, \quad \gamma = e^{-\delta} \tag{2}$$

The assumption is that in the initial stages of learning past knowledge is more important, over time as the model specializes on a certain task we rely less on incorporating prior knowledge over time. In its simplest form, this can be represented as a moving average over the development set error curve so to choose δ as shown in Equation 3, where k is the size of the sliding window. In some cases an average over time is not suitable when the training error is volatile between slope estimations. Hence, alternative smoothing approaches would include kernel and spline models (Eubank, 1999) for fitting noisy, or volatile error curves.

$$\delta_t = \mathbb{E}[\nabla_{[t,t+k]}] \tag{3}$$

A kernel ψ can be used to smooth over the error curve, which can take the form of a Gaussian kernel $\psi(\hat{x}, x_i) = e^{\left(-(\hat{x}-x_i)^2/2b^2\right)}$. Another approach is to use Local Weighted Scatterplot Smoothing (LOWESS) (Cleveland, 1979; Cleveland and Devlin, 1988) which is a non-parametric regression technique that is more robust against outliers in comparison to standard least square regression by adding a penalty term. Equation 4 shows the regularized least squares function for a set of cubic smoothing splines ψ which are piecewise polynomials that are connected by *knots*, commonly distributed uniformly across the given interval $[0, T]$. Splines are solved using least squares with a regularization term $\lambda\theta_j^2 \; \forall \; j$ as shown in Equation 4, ψ_j being a single piecewise polynomial at the subinterval $[t, t+n] \in [0, T]$. Each subinterval represents the space that γ is adapted for over time i.e change the influence of the $\mathcal{T}_s$ *Dropping* Network as $\mathcal{T}_t$ model learns from few examples over time. This type of cubic spline is used for the subsequent result section for *Dropping* Network transfer.

$$\hat{\delta}_{[t]} = \arg\min_{\theta} \sum_{i=1}^{n} \Big(y_i - \sum_{j=1}^{J} \theta_j \psi_j(x_i) \Big)^2 + \lambda \sum_{j=1}^{J} \theta_j^2 \tag{4}$$

.

Classification is then carried out using standard Cross-Entropy (CE) loss as shown in Equation 5.

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{M} y_{i,c} \log(\hat{y}_{i,c}) \tag{5}$$

This approach is relatively straightforward and on average across all three datasets, 58% more computational time for training 10 smaller ensembles for each single-task was needed, in comparison to a larger global model on a single NVIDIA Quadro M2000 Graphic Processing Unit.

Some benefits of the proposed method can be noted at this point. Firstly, the distance measure to related tasks is directly proportional to the online error of the target task. In contrast, hard parameter sharing does not address such issues. Although, there has also been recent approaches that use Gaussian Kernel Density estimates to initialize the parameters where uncertainty is accounted for (O' Neill and Buitelaar, 2018). Although, not the focus of this work, the $\mathcal{T}_t$ model can be trained on a new task with more or less classes by adding or discarding connections on the last

softmax layer. Secondly, by weighting the models within the ensemble that perform better on $\mathcal{T}_t$ we mitigate *negative transfer* problems. We now discuss some of the main results of the proposed *Dropping* Network transfer.

5 Results

The evaluation is carried out on both the rate of convergence and optimal performance. Hence, we particularly analyze the speedup obtained in the early stages of learning. Table 1 shows the results on all three datasets for single-task learning, the purpose of which is to clarify the potential performance if learned from most of the available training data (between 70%-80% of the overall dataset for the three datasets). The ensemble model slightly outperforms other networks proposed, while the co-attention network produces similar performance with a similar architecture to the ensemble models except for the use of local attention over hidden layers shared across both sentences. The improvements are most notable on MNLI, reaching competitive performance in comparison to state of the art (SoTA) on the RepEval task[2], held by Chen et al. (Chen et al., 2017c) which similarly uses a Gated Attention Network. These SoTA results are considered as an upper bound to the potential performance when evaluating the *Dropping* based TL strategy for few shot learning.

Figure 2 demonstrates the performance of the zero-shot learning results of the ensemble network which averages the probability estimates from each models prediction on the $\mathcal{T}_t$ test set (few-shot $\mathcal{T}_t$ training set or development set not included). As the ensembles learn on $\mathcal{T}_s$ it is evident that most of the learning has already been carried out by 5000-10,000 epochs. Producing entailment and contradiction predictions for multi-genre sources is significantly more difficult, demonstrated by lower test accuracy when transferring SNLI $\rightarrow$ MNLI, in comparison to MNLI $\rightarrow$ SNLI that performs better relative to recent SoTA on SNLI.

Table 2 shows a comparison of the transfer methods. The first approach is straightforward hard parameter transfer the results from transferring parameters from the *Dropping* network trained with the output shown in Equation 2. The ensemble consists of 10 smaller network on $\mathcal{T}_t$ with a dropout rate $p_d = 0.5$. In the case with SNLI + QM (ie. SNLI + Question Matching) and MNLI + QM, 20 ensembles are transferred, 10 from each model. The QM dataset is not as "similar" in nature and in the zero-shot learning setting the model's weights a_S and a_Q are normalized to 1 (however, this could have been weighted based on a prior belief of how "similar" the tasks are). Hence, it is unsurprising that QM dataset has reduced the test accuracy given it is further to $\mathcal{T}_t$ than S is. The second approach displays the baseline few-shot learning performance with fixed parameter transferred from $\mathcal{T}_t$ on the lower layer with fine-tuning of the 2^{nd} layer. Here, we ensure that instances from each genre within MNLI are sampled at least 100 times and that the batch of 3% the original size of the corpus is used (14,000 instances). Since SNLI and QM are created from a single source, we did not to impose such a constraint, also using a 3% random sample for testing. Therefore, these results and all subsequent results denoted as *Train Acc.* % refers to the training accuracy on the small batches for each respective dataset. We see improvements that are made from further tuning on the small $\mathcal{T}_t$ batch that are made, particularly on MNLI with a 2.815 percentage point increase in test accuracy. For both SNLI + QM $\rightarrow$ MNLI and MNLI + QM $\rightarrow$ SNLI cases final predictions are made by averaging over the class probability estimates before using CE loss.

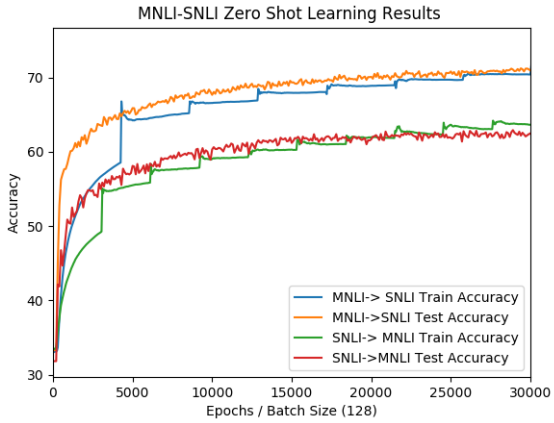

Figure 2: *Zero-Shot Learning Between NLU Tasks*

The third method is the proposed model that transfers parameters from the *Dropping* network trained with the output seen in Equation 2, using a spline smoother with piecewise polynomials (as described in Equation 4). This approach finds the slope of the online error curve between sub-intervals so to choose γ i.e the balance between

[2] https://repeval2017.github.io/shared/

	MNLI				SNLI				QM			
	Train		Test		Train		Test		Train		Test	
	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL
GRU-1h	91.927	0.230	68.420	1.112	89.495	0.233	77.347	0.755	84.577	0.214	78.898	0.389
GRU-2h	90.439	0.243	68.277	1.121	89.464	0.224	79.628	0.626	86.308	0.096	77.059	0.092
Bi-GRU-2h	90.181	0.253	68.716	1.065	89.703	0.226	80.594	0.636	88.011	0.108	77.522	0.267
Co-Attention GRU-2h	94.341	0.183	70.692	0.872	91.338	0.211	82.513	0.583	89.690	0.088	81.550	0.218
Ensemble Bi-GRU-2h	91.767	0.260	70.748	0.829	90.091	0.218	81.650	0.492	88.481	0.177	83.820	0.194

Table 1: Single Task Compositional Similarity Learning Results (shaded values represent best performing models)

	Zero-Shot Hard Parameter Transfer				*Few-Shot* Transfer Learning				Dropping-GRU CSES			
	Train		Test		Train		Test		Train		Test	
	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL	Acc. / %	LL
S $\rightarrow$ M	60.439	0.243	61.277	1.421	89.655	0.248	64.897	1.696	90.439	0.243	66.207	1.721
S+Q $\rightarrow$ M	62.317	0.208	62.403	1.392	87.014	0.376	65.218	1.255	86.649	0.317	70.703	0.576
M $\rightarrow$ S	74.609	0.611	71.662	0.844	86.445	0.260	73.141	0.729	90.181	0.253	72.716	0.615
M+Q $\rightarrow$ S	74.911	0.603	68.006	0.924	85.922	0.281	70.541	0.911	91.783	0.228	77.926	0.598

Table 2: *Zero-Shot* Hard Parameter Transfer (left), Few-Shot Transfer Learning with Fixed Lower Hidden GRU-Layer Parameter Transfer From $\mathcal{T}_s$ and Fine-Tuned Upper Layer Trained On $\mathcal{T}_t$ (middle) and Dropping-GRU Cubic Spline Error Smoothing (right)

the source ensemble and target model trained on few examples. The ensemble consists of 20 smaller networks on $\mathcal{T}_t$ with a dropout rate $p_d = 0.5$. We note that unlike the previous two baselines method shown in Table 2, the performance does not decrease by transferring the QM models to both SNLI and MultiNLI. This is explained by the use of the weighting scheme proposed with spline smoothing of the error curve i.e γ decreases at a faster rate for $\mathcal{T}_t$ due to the ineffectiveness of the ensembles created on the QM dataset. In summary, we find transfer of MNLI + QM $\rightarrow$ SNLI and SNLI+QM $\rightarrow$ MNLI showing most improvement using the proposed transfer method, in comparison to standard hard and soft parameter transfer. This is reflected in the fact that the proposed method is the only one which improved on SNLI while still transferring the more distant QM dataset.

6 Conclusion

Dropping Networks are based on a simple notion that combines two common meta-learning model averaging methods: *Bagging* and *Dropout*. The combination of both can be of benefit to overcome some limitations in transfer learning such as learning from more distant tasks and mitigating *negative transfer*, most interestingly, in the few-shot learning setting. This paper has empirically demonstrated this on learning complex semantic relationships between sentence pairs. Additionally, We find the co-attention network and the ensemble GRU network to perform comparably for single-task learning. Below we summarize some of the main points and findings from this work:

- The method for transfer only relies on one additional parameter γ. Also, using a higher decay rate γ (0.9-0.95) is more suitable for tasks that are closely related.
- Decreasing γ in proportion to the slope of a smooth spline fitted to the online error curve performs better than arbitrary step changes or a fixed rate for γ (equivalent to static hard parameter ensemble transfer).

Finally, the proposed transfer procedures using *Dropping* networks has been demonstrated in the context of natural language, although the method is applicable to any standard, spatial or recurrent-based neural network.

References

First quora dataset release: Question pairs. https://data.quora.com/First-Quora\-Dataset-Release-Question-Pairs. Posted: 2017-01-24.

Pierre Baldi and Peter Sadowski. 2014. The dropout learning algorithm. *Artificial intelligence*, 210:78–122.

Pierre Baldi and Peter J Sadowski. 2013. Understanding dropout. In *Advances in neural information processing systems*, pages 2814–2822.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, and Diana Inkpen. 2017a. Natural language inference with external knowledge. *arXiv preprint arXiv:1711.04289*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Enhanced lstm for natural language inference. In *Proc. ACL*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017c. Recurrent neural network-based sentence encoder with gated attention for natural language inference. *arXiv preprint arXiv:1708.01353*.

William S Cleveland. 1979. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836.

William S Cleveland and Susan J Devlin. 1988. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403):596–610.

Randall L Eubank. 1999. *Nonparametric regression and spline smoothing*. CRC press.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

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

Bill MacCartney. 2009. *Natural language inference*. Stanford University.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 1–8.

Llew Mason, Jonathan Baxter, Peter L Bartlett, and Marcus R Frean. 2000. Boosting algorithms as gradient descent. In *Advances in neural information processing systems*, pages 512–518.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2015. Natural language inference by tree-based convolution and heuristic matching. *arXiv preprint arXiv:1512.08422*.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How transferable are neural networks in nlp applications? *arXiv preprint arXiv:1603.06111*.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI*, pages 2786–2792.

Nikita Nangia, Adina Williams, Angeliki Lazaridou, and Samuel R Bowman. 2017. The repeval 2017 shared task: Multi-genre natural language inference with sentence representations. *arXiv preprint arXiv:1707.08172*.

James O' Neill and Paul Buitelaar. 2018. Few shot transfer learning between term relatedness and similarity tasks using a gated recurrent siamese network. In *AAAI*.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Lorien Y Pratt. 1993. Discriminability-based transfer between neural networks. In *Advances in neural information processing systems*, pages 204–211.

Lorien Y Pratt, Jack Mostow, Candace A Kamm, and Ace A Kamm. 1991. Direct transfer of learned information among neural networks. In *AAAI*, volume 91, pages 584–589.

Rajat Raina, Andrew Y Ng, and Daphne Koller. 2006. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 713–720. ACM.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Yikang Shen, Wenge Rong, Nan Jiang, Baolin Peng, Jie Tang, and Zhang Xiong. 2017. Word embedding based correlation model for question/answer matching. In *AAAI*, pages 3511–3517.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017a. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017b. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

David Warde-Farley, Ian J Goodfellow, Aaron Courville, and Yoshua Bengio. 2013. An empirical analysis of dropout in piecewise linear networks. *arXiv preprint arXiv:1312.6197*.

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

Han Yang, Marta R Costa-jussà, and José AR Fonollosa. 2017. Character-level intra attention network for natural language inference. *arXiv preprint arXiv:1707.07469*.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

Liang Zheng, Yali Zhao, Shengjin Wang, Jingdong Wang, and Qi Tian. 2016. Good practice in cnn feature transfer. *arXiv preprint arXiv:1604.00133*.