Abstract

It can be challenging to train multi-task neural networks that outperform or even match their single-task counterparts. To help address this, we propose using knowledge distillation where single-task models teach a multi-task model. We enhance this training with teacher annealing, a novel method that gradually transitions the model from distillation to supervised learning, helping the multi-task model surpass its single-task teachers. We evaluate our approach by multi-task fine-tuning BERT on the GLUE benchmark. Our method significantly improves over standard single-task and multi-task training, resulting in state-of-the-art accuracies.

1 Introduction

Building a single model that jointly learns to perform many tasks effectively has been a longstanding challenge in NLP. However, multi-task NLP remains ineffective for many applications, with multi-task models often performing worse than their single-task counterparts (Plank and Alonso, 2017; Bingel and Søgaard, 2017; McCann et al., 2018). Motivated by these results, we propose a way of applying knowledge distillation (Hinton et al., 2015) so that single-task models effectively teach a multi-task model.

Knowledge distillation transfers knowledge from a “teacher” model to a “student” model by training the student to imitate the teacher’s predictions. In “born-again networks” (Furlanello et al., 2018), the teacher and student models have the same neural architecture and model size, but surprisingly the student is able to surpass the teacher’s accuracy. Our work extends these ideas to the multi-task setting. We compare Single→Multi1 born-again distillation with several other variants (Single→Single and Multi→Multi), and also explore performing multiple rounds of distillation (Single→Multi→Single→Multi). Furthermore, we propose a simple teacher annealing method that helps the student model outperform its teachers, substantially improving results.

Our experiments build upon recent success in unsupervised pre-training (Dai and Le, 2015; Peters et al., 2018) and multi-task fine-tune BERT (Devlin et al., 2018) to perform the tasks from the GLUE natural language understanding benchmark (Wang et al., 2019). Our training method, which we call Born-Again Multi-tasking (BAM)2, consistently outperforms standard single-task and multi-task training, resulting in state-of-the-art accuracies on GLUE. Further analysis shows the multi-task models benefit from both better regularization and transfer between related tasks.

2 Related Work

Multi-task learning for neural networks in general (Caruana, 1997) and within NLP specifically (Collobert and Weston, 2008; Luong et al., 2016) has been widely studied. Much of the recent work for NLP has centered on neural architecture design: e.g., ensuring only beneficial information is shared

---

1We use Single→Multi to indicate distilling single-task “teacher” models into a multi-task “student” model.

2Code will be released at http://anonymized
across tasks (Liu et al., 2017; Ruder et al., 2019) or arranging tasks in linguistically-motivated hierarchies (Søgaard and Goldberg, 2016; Hashimoto et al., 2017; Sanh et al., 2019). These contributions are orthogonal to ours because we instead focus on the multi-task training algorithm.

Knowledge distillation has been used to regularize multi-task reinforcement learning agents (Parisotto et al., 2016; Teh et al., 2017). More relevant to this work, Tan et al. (2019) distill single-language-pair machine translation systems into a many-language system. However, they focus on multilingual rather than multi-task learning, use a more complex training procedure, and only experiment with Single→Multi distillation.

Concurrently with our work, Phang et al. (2018) and Liu et al. (2019) also explore fine-tuning BERT using multiple tasks. However, they use only standard transfer or multi-task learning, instead focusing on finding beneficial task pairs or augmenting the model with more sophisticated task-specific components.

3 Methods

3.1 Multi-Task Setup

Model. All of our models are built on top of BERT (Devlin et al., 2018). This model passes byte-pair-tokenized (Sennrich et al., 2016) input sentences through a Transformer network (Vaswani et al., 2017), producing a contextualized representation across tasks (Devlin et al., 2018). For multi-task training, examples of different tasks are shuffled together, even within minibatches. The (unweighted) summed loss across all tasks is minimized.

3.2 Knowledge Distillation

We use $D_\tau = (x_1^\tau, y_1^\tau), \ldots, (x_N^\tau, y_N^\tau)$ to denote the training set for a task $\tau$ and $p_\tau(y|x, \theta)$ to denote the probability distribution over classes for task $\tau$ produced by a neural network with parameters $\theta$ on the input $x$. Standard supervised learning trains the model $\theta$ to minimize the cross-entropy loss on the training set:

$$L(\theta) = \sum_{x^\tau, y^\tau \in D_\tau} CE(y^\tau, p_\tau(y|x^\tau, \theta))$$

Knowledge distillation trains the model to instead match the predictions of a teacher model with parameters $\theta'$:

$$L(\theta) = \sum_{x^\tau, y^\tau \in D_\tau} CE(p_\tau(y|x^\tau, \theta'), p_\tau(y|x^\tau, \theta))$$

Note that our distilled networks are “born-again” in that the student has the same model architecture as the teacher, i.e., all of our models have the same prediction function $p_\tau$ for each task. For regression tasks, we train the student to minimize the L2 distance between its prediction and the teacher’s instead of using cross-entropy loss. Intuitively, knowledge distillation improves training because the full distribution over labels provided by the teacher provides a richer training signal than a one-hot label. See Furlanello et al. (2018) for a more thorough discussion.

Multi-Task Distillation. Given a set of tasks $T$, we train a single-task model with parameters $\theta_\tau$ on each task $\tau$. For most experiments, we use the single-task models to teach a multi-task model:

$$L(\theta) = \sum_{\tau \in T} \sum_{x^\tau, y^\tau \in D_\tau} CE(p_\tau(y|x^\tau, \theta_\tau), p_\tau(y|x^\tau, \theta))$$

However, we experiment with other distillation strategies as well.

Teacher Annealing. In knowledge distillation, the student is trained to imitate the teacher. This raises the concern that the student may be limited by the teacher’s performance and not be able to substantially outperform the teacher. To address this, we propose teacher annealing, which mixes the teacher prediction with the gold label during training. Specifically, the cross-entropy term in the summation becomes

$$CE(\lambda y^\tau + (1 - \lambda)p_\tau(y|x^\tau, \theta_\tau), p_\tau(y|x^\tau, \theta))$$

For BERT this is a special token [CLS] that is prepended to each input sequence.

We normalize the labels so they are between 0 and 1.

For BERT code and weights, see https://github.com/google-research/bert.
Table 1: Comparison of methods on the GLUE dev set. *, **, and *** indicate statistically significant \((p < .05, p < .01, \text{ and } p < .001)\) improvements over both Single and Multi according to bootstrap hypothesis tests.\(^6\)

| Model      | Avg. \(|D| = 8.5k\) | CoLA 67k | SST-2 3.7k | MRPC 5.8k | STS-B 364k | QQP 393k | MNLI 108k | QNLI 2.5k | RTE 70.4 |
|------------|----------------------|---------|-----------|-----------|-----------|---------|---------|--------|--------|
| Single     | 84.0                 | 60.6    | 93.2      | 88.0      | 90.0      | 91.3    | 86.6    | 92.3   | 70.4   |
| Multi      | 85.5                 | 60.3    | 93.3      | 88.0      | 89.8      | 91.4    | 86.5    | 92.2   | 82.1   |
| Single→Single | 84.3             | **61.7** | 93.2      | **88.7**  | 90.0      | 91.4    | **86.8** | **92.5** | 70.0   |
| Multi→Multi | 85.6                 | 60.9    | 93.5      | 88.1      | 89.8      | **91.5** | 86.7    | 92.3   | 82.0   |
| Single→Multi | **86.0**             | **61.8** | **93.6**  | **89.3**  | **89.7**  | **91.6** | **87.0** | **92.5** | **82.8** |

where \(\lambda\) is linearly increased from 0 to 1 throughout training. Early in training, the model is mostly distilling to get as useful of a training signal as possible. Towards the end of training, the model is mostly relying on the gold-standard labels so it can learn to surpass its teachers.

### 4 Experiments

**Data.** We use the GLUE dataset (Wang et al., 2019), which consists of 9 natural language understanding tasks covering textual entailment (RTE and MNLI) question-answer entailment (QNLI), paraphrase (MRPC), question paraphrase (QQP), textual similarity (STS), sentiment (SST-2), linguistic acceptability (CoLA), and Winograd Schema (WNLI).

**Training Details.** Rather than simply shuffling the datasets for our multi-task models, we follow the task sampling procedure from Bowman et al. (2018), where the probability of training on an example for a particular task \(\tau\) is proportional to \(|D_\tau|^{0.75}\). This ensures that tasks with very large datasets don’t overly dominate the training.

We also use the layerwise-learning-rate-trick from Howard and Ruder (2018). If layer 0 is the NN layer closest to the output, the learning rate for a particular layer \(d\) is set to \(\text{BASE}_\text{LR} \cdot \alpha^d\) (i.e., layers closest to the input get lower learning rates). The intuition is that pre-trained layers closer to the input learn more general features, so they shouldn’t be altered much during training.

**Hyperparameters.** For single-task models, we use the same hyperparameters as in the original BERT experiments except we pick a layerwise-learning-rate decay \(\alpha\) of 1.0 or 0.9 on the dev set for each task. For multi-task models, we train the model for longer (6 epochs instead of 3) and with a larger batch size (128 instead of 32), using \(\alpha = 0.9\) and a learning rate of 1e-4. All models use the BERT-Large pre-trained weights.

**Reporting Results.** Dev set results report the average score\(^7\) on all GLUE tasks except WNLI, for which methods can’t outperform a majority baseline. All results show the median score of at least 20 trials with different random seeds. We find using a large number of trials is essential because results can vary significantly for different runs. For example, standard deviations in score are over \(\pm 1\) for CoLA, RTE, and MRPC for multi-task models. Single-task standard deviations are even larger.

### 5 Results

**Main Results.** We compare models trained with single-task learning, multi-task learning, and several varieties of distillation in Table 1. While standard multi-task training improves over single-task training for RTE (likely because it is closely related to MNLI), there is no improvement on the other tasks. In contrast, Single→Multi knowledge distillation improves or matches the performance of the other methods on all tasks except STS, the only regression task in GLUE. We believe distillation does not work well for regression tasks because there is no distribution over classes passed on by the teacher to aid learning.

The gain for Single→Multi over Multi is larger than the gain for Single→Single over Single, suggesting that distillation works particularly well in combination with multi-task learning. Interestingly, Single→Multi works substantially better than Multi→Multi distillation. We speculate it may help that the student is exposed to a diverse set of teachers in the same way ensembles bene-

\(^6\)For all statistical tests we use the Holm-Bonferroni method (Holm, 1979) to correct for multiple comparisons.

\(^7\)Spearman correlation for STS, Matthews correlation for CoLA, and accuracy for the other tasks.
Table 2: Comparison of test set results.

<table>
<thead>
<tr>
<th>Model</th>
<th>GLUE score</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT-Base (Devlin et al., 2018)</td>
<td>78.5</td>
</tr>
<tr>
<td>BERT-Large (Devlin et al., 2018)</td>
<td>80.5</td>
</tr>
<tr>
<td>BERT on STILTs (Phang et al., 2018)</td>
<td>82.0</td>
</tr>
<tr>
<td>MT-DNN (Liu et al., 2019)</td>
<td>82.2</td>
</tr>
<tr>
<td>BERT-Large + BAM (ours)</td>
<td>82.3</td>
</tr>
</tbody>
</table>

Table 3: Ablation Study. Differences from Single→Multi are statistically significant ($p < .001$) according to Mann-Whitney U tests.6

<table>
<thead>
<tr>
<th>Trained Tasks</th>
<th>RTE score</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTE</td>
<td>70.0</td>
</tr>
<tr>
<td>RTE + MNLI</td>
<td>83.4</td>
</tr>
<tr>
<td>RTE + QQP + CoLA + SST</td>
<td>75.1</td>
</tr>
<tr>
<td>All GLUE</td>
<td>82.8</td>
</tr>
</tbody>
</table>

Table 4: Which tasks help RTE? Pairwise differences are statistically significant ($p < .01$) according to Mann-Whitney U tests.5

<table>
<thead>
<tr>
<th>Trained Tasks</th>
<th>RTE score</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTE + MNLI</td>
<td>83.4</td>
</tr>
<tr>
<td>RTE + QQP + CoLA + SST</td>
<td>75.1</td>
</tr>
<tr>
<td>All GLUE</td>
<td>82.8</td>
</tr>
</tbody>
</table>

Discussion and Conclusion

We have shown that Single→Multi distillation combined with teacher annealing produces results consistently better than standard single-task or multi-task training. Achieving robust multi-task gains across many tasks has remained elusive in previous research, so we hope our work will make multi-task learning more broadly useful within NLP. However, with the exception of closely related tasks with small datasets (e.g., MNLI helping RTE), the overall size of the gains from our multi-task method are small compared to the gains provided by transfer learning from unsupervised tasks (i.e., BERT). It remains to be fully understood to what extent “unsupervised pre-training is all you need” and where transfer/multi-task learning from supervised tasks can provide the most value.
References


Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In NIPS.


Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In ICLR.


Barbara Plank and Héctor Martínez Alonso. 2017. When is multitask learning effective? Semantic sequence prediction under varying data conditions. In EACL.


Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In ACL.


