What can we learn from Semantic Tagging?

Abstract

We investigate the effects of multi-task learning using the recently introduced task of semantic tagging. We employ semantic tagging as an auxiliary task for three different NLP tasks: part-of-speech tagging, Universal Dependency parsing, and Natural Language Inference. We compare full neural network sharing, partial neural network sharing, and what we term the learning what to share setting where negative transfer between tasks is less likely. Our findings show considerable improvements for all tasks, particularly in the learning what to share setting, which shows consistent gains across all tasks.

Introduction

Multi-task learning (MTL) is a recently resurgent approach to machine learning in which multiple tasks are simultaneously learned. By optimising the multiple loss functions of related tasks at once, multi-task learning models can achieve superior results compared to models trained on a single task. The key principle is summarized by BIBREF0 as “MTL improves generalization by leveraging the domain-specific information contained in the training signals of related tasks". Neural MTL has become an increasingly successful approach by exploiting similarities between Natural Language Processing (NLP) tasks BIBREF1 , BIBREF2 , BIBREF3 . Our work builds upon BIBREF4 , who demonstrate that employing semantic tagging as an auxiliary task for Universal Dependency BIBREF5 part-of-speech tagging can lead to improved performance.

The objective of this paper is to investigate whether learning to predict lexical semantic categories can be beneficial to other NLP tasks. To achieve this we augment single-task models (ST) with an additional classifier to predict semantic tags and jointly optimize for both the original task and the auxiliary semantic tagging task. Our hypothesis is that learning to predict semantic tags as an auxiliary task can improve performance of single-task systems. We believe that this is, among other factors, due to the following:

We test our hypothesis on three disparate NLP tasks: (i) Universal Dependency part-of-speech tagging (UPOS), (ii) Universal Dependency parsing (UD DEP), a complex syntactic task; and (iii) Natural Language Inference (NLI), a complex task requiring deep natural language understanding.

Semantic Tagging

Semantic tagging BIBREF4 , BIBREF7 is the task of assigning language-neutral semantic categories to words. It is designed to overcome a lack of semantic information syntax-oriented part-of-speech tagsets, such as the Penn Treebank tagset BIBREF8 , usually have. Such tagsets exclude important semantic distinctions, such as negation and modals, types of quantification, named entity types, and the contribution of verbs to tense, aspect, or event.

The semantic tagset is language-neutral, abstracts over part-of-speech and named-entity classes, and includes fine-grained semantic information. The tagset consists of 80 semantic tags grouped in 13 coarse-grained classes. The tagset originated in the Parallel Meaning Bank (PMB) project BIBREF9 , where it contributes to compositional semantics and cross-lingual projection of semantic representations. Recent work has highlighted the utility of the tagset as a conduit for evaluating the semantics captured by vector representations BIBREF10 , or employed it in an auxiliary tagging task BIBREF4 , as we do in this work.

Learning What to Share

Recently, there has been an increasing interest in the development of models which are trained to learn what to (and what not to) share between a set of tasks, with the general aim of preventing negative transfer when the tasks are not closely related BIBREF11 , BIBREF12 , BIBREF13 , BIBREF14 . Our Learning What to Share setting is based on this idea and closely related to BIBREF15 's shared layer architecture.

Specifically, a layer $\vec{h}_{X}$ which is shared between the main task and the auxiliary task is split into two subspaces: a shared subspace $\vec{h}_{X_{S}}$ and a private subspace $\vec{h}_{X_{P}}$. The interaction between the shared subspaces is modulated via a sigmoidal gating unit applied to a set of learned weights, as seen in Equations (9) and () where $\vec{h}_{X_{S(main)}}$ and $\vec{h}_{X_{S(aux)}}$ are the main and auxiliary tasks' shared layers, $W_{a\rightarrow m}$ and $W_{m\rightarrow a}$ are learned weights, and $\sigma $ is a sigmoidal function.

$$\vec{h}_{X_{S(main)}} &= \vec{h}_{X_{S(main)}} \sigma (\vec{h}_{X_{S(aux)}} W_{a\rightarrow m})\\ \vec{h}_{X_{S(aux)}} &= \vec{h}_{X_{S(aux)}} \sigma (\vec{h}_{X_{S(main)}} W_{m\rightarrow a})$$ (Eq. 9)

Unlike BIBREF15 's Shared-Layer Architecture, in our setup each task has its own shared subspace rather than one common shared layer. This enables the sharing of different parameters in each direction (i.e., from main to auxiliary task and from auxiliary to main task), allowing each task to choose what to learn from the other, rather than having “one shared layer to capture the shared information for all the tasks” as in BIBREF15 .

Multi-Task Learning Settings

We implement three neural MTL settings, shown in Figure 1 . They differ in the way the network's parameters are shared between the tasks:

Data

In the UPOS tagging experiments, we utilize the UD 2.0 English corpus BIBREF16 for the POS tagging and the semantically tagged PMB release 0.1.0 (sem-PMB) for the MTL settings. Note that there is no overlap between the two datasets. Conversely, for the UD DEP and NLI experiments there is a complete overlap between the datasets of main and auxiliary tasks, i.e., each instance is labeled with both the main task's labels and semantic tags. We use the Stanford POS Tagger BIBREF17 trained on sem-PMB to tag the UD corpus and NLI datasets with semantic tags, and then use those assigned tags for the MTL settings of our dependency parsing and NLI models. We find that this approach leads to better results when the main task is only loosely related to the auxiliary task. The UD DEP experiments use the English UD 2.0 corpus, and the NLI experiments use the SNLI BIBREF18 and SICK-E datasets BIBREF19 . The provided train, development, and test splits are used for all datasets. For sem-PMB, the silver and gold parts are used for training and testing respectively.

Experiments

We run four experiments for each of the four tasks (UPOS, UD DEP, SNLI, SICK-E), one using the ST model and one for each of the three MTL settings. Each experiment is run five times, and the average of the five runs is reported. We briefly describe the ST models and refer the reader to the original work for further details due to a lack of space. For reproducibility, detailed diagrams of the MTL models for each task and their hyperparameters can be found in Appendix "MTL setting Diagrams, Preprocessing, and Hyperparameters" .

Universal Dependency POS Tagging

Our tagging model uses a basic contextual one-layer bi-LSTM BIBREF20 that takes in word embeddings and produces a sequence of recurrent states which can be viewed as contextualized representations. The recurrent r_n state from the bi-LSTM corresponding to each time-step t_n is passed through a dense layer with a softmax activation to predict the token's tag.

In each of the MTL settings a softmax classifier is added to predict a token's semantic tag and the model is then jointly trained on the concatenation of the sem-PMB and UPOS tagging data to minimize the sum of softmax cross-entropy losses of both the main (UPOS tagging) and auxiliary (semantic tagging) tasks.

Universal Dependency Parsing

We employ a parsing model that is based on BIBREF21 BIBREF21 . The model's embeddings layer is a concatenation of randomly initialized word embeddings and character-based word representations added to pre-trained word embeddings, which are passed through a 4-layer stacked bi-LSTM. Unlike BIBREF21 , our model jointly learns to perform UPOS tagging and parsing, instead of treating them as separate tasks. Therefore, instead of tag embeddings, we add a softmax classifier to predict UPOS tags after the first bi-LSTM layer. The outputs from that layer and the UPOS softmax prediction vectors are both concatenated to the original embedding layer and passed to the second bi-LSTM layer. The output of the last bi-LSTM is then used as input for four dense layers with a ReLU activation, producing four vector representations: a word as a dependent seeking its head; a word as a head seeking all its dependents; a word as a dependent deciding on its label; a word as head deciding on the labels of its dependents. These representations are then passed to biaffine and affine softmax classifiers to produce a fully-connected labeled probabilistic dependency graph BIBREF21 . Finally, a non-projective maximum spanning tree parsing algorithm BIBREF22 , BIBREF23 is used to obtain a well-formed dependency tree.

Similarly to UPOS tagging, an additional softmax classifier is used to predict a token's semantic tag in each of the MTL settings, as both tasks are jointly learned. In the FSN setting, the 4-layer stacked bi-LSTM is entirely shared. In the PSN setting the semantic tags are predicted from the second layer's hidden states, and the final two layers are devoted to the parsing task. In the LWS setting, the first two layers of the bi-LSTM are split into a private bi-LSTM $_{private}$ and a shared bi-LSTM $_{shared}$ for each of the tasks with the interaction between the shared subspaces being modulated via a gating unit. Then, two bi-LSTM layers that are devoted to parsing only are stacked on top.

Natural Language Inference

We base our NLI model on BIBREF25 's Enhanced Sequential Inference Model which uses a bi-LSTM to encode the premise and hypothesis, computes a soft-alignment between premise and hypothesis' representations using an attention mechanism, and employs an inference composition bi-LSTM to compose local inference information sequentially. The MTL settings are implemented by adding a softmax classifier to predict semantic tags at the level of the encoding bi-LSTM, with rest of the model unaltered. In the FSN setting, the hidden states of the encoding bi-LSTM are directly passed as input to the softmax classifier. In the PSN setting an earlier bi-LSTM layer is used to predict the semantic tags and the output from that is passed on to the encoding bi-LSTM which is stacked on top. This follows BIBREF26 's hierarchical approach. In the LWS setting, a bi-LSTM layer with private and shared subspaces is used for semantic tagging and for the ESIM model's encoding layer. In all MTL settings, the bi-LSTM used for semantic tagging is pre-trained on the sem-PMB data.

Results and Discussion

Results for all tasks are shown in Table 1 . In line with BIBREF4 's findings, the FSN setting leads to an improvement for UPOS tagging. POS tagging, a sequence labeling task, can be seen as the most closely related to semantic tagging, therefore negative transfer is minimal and the full sharing of parameters is beneficial. Surprisingly, the FSN setting also leads improvements for UD DEP. Indeed, for UD DEP, all of the MTL models outperform the ST model by increasing margins. For the NLI tasks, however, there is a clear degradation in performance.

The PSN setting shows mixed results and does not show a clear advantage over FSN for UPOS and UD DEP. This suggests that adding task-specific layers after fully-shared ones does not always enable sufficient task specialization. For the NLI tasks however, PSN is clearly preferable to FSN, especially for the small-sized SICK-E dataset where the FSN model fails to adequately learn.

As a sentence-level task, NLI is functionally dissimilar to semantic tagging. However, it is a task which requires deep understanding of natural language semantics and can therefore conceivably benefit from the signal provided by semantic tagging. Our results demonstrate that it is possible to leverage this signal given a selective sharing setup where negative transfer can be minimized. Indeed, for the NLI tasks, only the LWS setting leads to improvements over the ST models. The improvement is larger for the SICK-E task which has a much smaller training set and therefore stands to learn more from the semantic tagging signal. For all tasks, it can be observed that the LWS models outperform the rest of the models. This is in line with our expectations with the findings from previous work BIBREF12 , BIBREF15 that selective sharing outperforms full network and partial network sharing.

Analysis

In addition to evaluating performance directly, we attempt to qualify how semtags affect performance with respect to each of the SNLI MTL settings.

Qualitative analyses

The fact that NLI is a sentence-level task, while semantic tags are word-level annotations presents a difficulty in measuring the effect of semantic tags on the systems' performance, as there is no one-to-one correspondence between a correct label and a particular semantic tag. We therefore employ the following method in order to assess the contribution of semantic tags. Given the performance ranking of all our systems — $FSN < ST < PSN < LWS$ — we make a pairwise comparison between the output of a superior system S_{sup} and an inferior system S_{inf} . This involves taking the pairs of sentences that every S_{sup} classifies correctly, but some S_{inf} does not. Given that FSN is the worst performing system and, as such, has no `worse' system for comparison, we are left with six sets of sentences: ST-FSN, PSN-FSN, PSN-ST, LWS-PSN, LWS-ST, and LWS-FSN. To gain insight as to where a given system S_{sup} performs better than a given S_{inf} , we then sort each comparison sentence set by the frequency of semtags predicted therein, which are normalized by dividing by their frequency in the full SNLI test set.

We notice interesting patterns, visible in Figure 2 . Specifically, PSN appears markedly better at sentences with named entities (ART, PER, GEO, ORG) and temporal entities (DOM) than both ST and the FSN. Marginal improvements are also observed for sentences with negation and reflexive pronouns. The LWS setting continues this pattern, with additional improvements observable for sentences with the HAP tag for names of events, SST for subsective attributes, and the ROL tag for role nouns.

Contribution of semantic tagging

To assess the contribution of the semantic tagging auxiliary task independent of model architecture and complexity we run three additional SNLI experiments — one for each MTL setting — where the model architectures are unchanged but the auxiliary tasks are assigned no weight (i.e. do not affect the learning). The results confirm our previous findings that, for NLI, the semantic tagging auxiliary task only improves performance in a selective sharing setting, and hurts it otherwise: i) the FSN system which had performed below ST improves to equal it and ii) the PSN and LWS settings both see a drop to ST-level performance.

Conclusions

We present a comprehensive evaluation of MTL using a recently proposed task of semantic tagging as an auxiliary task. Our experiments span three types of NLP tasks and three MTL settings. The results of the experiments show that employing semantic tagging as an auxiliary task leads to improvements in performance for UPOS tagging and UD DEP in all MTL settings. For the SNLI tasks, requiring understanding of phrasal semantics, the selective sharing setup we term Learning What to Share holds a clear advantage. Our work offers a generalizable framework for the evaluation of the utility of an auxiliary task.

UPOS Tagging

fig:upos shows the three MTL models used for UPOS. All hyperparameters were tuned with respect to loss on the English UD 2.0 UPOS validation set. We trained for 20 epochs with a batch size of 128 and optimized using Adam BIBREF27 with a learning rate of 0.0001 . We weight the auxiliary semantic tagging loss with $\lambda $ = 0.1 . The pre-trained word embeddings we used are GloVe embeddings BIBREF28 of dimension 100 trained on 6 billion tokens of Wikipedia 2014 and Gigaword 5. We applied dropout and recurrent dropout with a probability of 0.3 to all bi-LSTMs.

UD DEP

fig:dep shows the three MTL models for UD DEP. We use the gold tokenization. All hyperparameters were tuned with respect to loss on the English UD 2.0 UD validation set. We trained for 15 epochs with a batch size of 50 and optimized using Adam with a learning rate of $2e-3$. We weight the auxiliary semantic tagging loss with $\lambda $ = 0.5 . The pre-trained word embeddings we use are GloVe embeddings of dimension 100 trained on 6 billion tokens of Wikipedia 2014 and Gigaword 5. We applied dropout with a probability of 0.33 to all bi-LSTM, embedding layers, and non-output dense layers.

NLI

fig:nli shows the three MTL models for NLI. All hyperparameters were tuned with respect to loss on the SNLI and SICK-E validation datasets (separately). For the SNLI experiments, we trained for 37 epochs with a batch size of 128. For the SICK-E experiments, we trained for 20 epochs with a batch size of 8. Note that the ESIM model was designed for the SNLI dataset, therefore performance is non-optimal for SICK-E. For both sets of experiments: we optimized using Adam with a learning rate of 0.00005 ; we weight the auxiliary semantic tagging loss with $\lambda $ = 0.1 ; the pre-trained word embeddings we use are GloVe embeddings of dimension 300 trained on 840 billion tokens of Common Crawl; and we applied dropout and recurrent dropout with a probability of 0.3 to all bi-LSTM, and non-output dense layers.

SNLI model output analysis

tab:examples shows demonstrative examples from the SNLI test set on which the Learning What to Share (LWS) model outperforms the single-task (ST) model. The examples cover all possible combinations of entailment classes. tab:semtags explains the relevant part of the semantic tagset. tab:fscore shows the per-label precision and recall scores.