Identifying beneficial task relations for multi-task learning in deep neural networks

Joachim Bingel
Department of Computer Science
University of Copenhagen
bingel@di.ku.dk

Anders Søgaard*
Department of Computer Science
University of Copenhagen
soegaard@di.ku.dk

Abstract

Multi-task learning (MTL) in deep neural networks for NLP has recently received increasing interest due to some compelling benefits, including its potential to efficiently regularize models and to reduce the need for labeled data. While it has brought significant improvements in a number of NLP tasks, mixed results have been reported, and little is known about the conditions under which MTL leads to gains in NLP. This paper sheds light on the specific task relations that can lead to gains from MTL models over single-task setups.

1 Introduction

Multi-task learning is receiving increasing interest in both academia and industry, with the potential to reduce the need for labeled data, and to enable the induction of more robust models. The main driver has been empirical results pushing state of the art in various tasks, but preliminary theoretical findings guarantee that multi-task learning works under various conditions. Some approaches to multi-task learning are, for example, known to work when the tasks share optimal hypothesis classes (Baxter, 2000) or are drawn from related sample generating distributions (Ben-David and Borberly, 2003).

In NLP, multi-task learning typically involves very heterogeneous tasks. However, while great improvements have been reported (Luong et al., 2016; Klerke et al., 2016), results are also often mixed (Collobert and Weston, 2008; Søgaard and Goldberg, 2016; Martínez Alonso and Plank, 2017), and theoretical guarantees no longer apply. The question *what task relations guarantee gains or make gains likely in NLP* remains open.

*Both authors contributed to the paper in equal parts.

Contributions This paper presents a systematic study of *when* and *why* MTL works in the context of sequence labeling with deep recurrent neural networks. We follow previous work (Klerke et al., 2016; Søgaard and Goldberg, 2016; Bollman and Søgaard, 2016; Plank, 2016; Braud et al., 2016; Martínez Alonso and Plank, 2017) in studying the set-up where hyperparameters from the single task architectures are reused in the multi-task setup (no additional tuning), which makes predicting gains feasible. Running MTL experiments on 90 task configurations and comparing their performance to single-task setups, we identify data characteristics and patterns in single-task learning that predict task synergies in deep neural networks. Both the LSTM code used for our single-task and multi-task models, as well as the script we used for the analysis of these, are available at github.com/jbingel/eacl2017_mtl.

2 Related work

In the context of structured prediction in NLP, there has been very little work on the conditions under which MTL works. Luong et al. (2016) suggest that it is important that the auxiliary data does not outsize the target data, while Benton et al. (2017) suggest that multi-task learning is particularly effective when we only have access to small amounts of target data. Martínez Alonso and Plank (2017) present a study on different task combinations with dedicated main and auxiliary tasks. Their findings suggest, among others, that success depends on how uniformly the auxiliary task labels are distributed.

Mou et al. (2016) investigate multi-task learning and its relation to transfer learning, and under which conditions these work between a set of sentence classification tasks. Their main finding with respect to multi-task learning is that success

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 164–169,
Valencia, Spain, April 3-7, 2017.

depends largely on "how similar in semantics the source and target datasets are", and that it generally bears close resemblance to transfer learning in the effect it has on model performance.

3 Multi-task Learning

While there are many approaches to multi-task learning, hard parameter sharing in deep neural networks (Caruana, 1993) has become extremely popular in recent years. Its greatest advantages over other methods include (i) that it is known to be an efficient regularizer, theoretically (Baxter, 2000), as well as in practice (Søgaard and Goldberg, 2016); and (ii) that it is easy to implement.

The basic idea in hard parameter sharing in deep neural networks is that the different tasks share some of the hidden layers, such that these learn a joint representation for multiple tasks. Another conceptualization is to think of this as regularizing our target model by doing model interpolation with auxiliary models in a dynamic fashion.

Multi-task linear models have typically been presented as matrix regularizers. The parameters of each task-specific model makes up a row in a matrix, and multi-task learning is enforced by defining a joint regularization term over this matrix. One such approach would be to define the joint loss as the sum of losses and the sum of the singular values of the matrix. The most common approach is to regularize learning by the sum of the distances of the task-specific models to the model mean. This is called mean-constrained learning. Hard parameter sharing can be seen as a very crude form of mean-constrained learning, in which parts of all models (typically the hidden layers) are enforced to be identical to the mean.

Since we are only forcing parts of the models to be identical, each task-specific model is still left with wiggle room to model heterogeneous tasks, but the expressivity is very limited, as evidenced by the inability of such networks to fit random noise (Søgaard and Goldberg, 2016).

3.1 Models

Recent work on multi-task learning of NLP models has focused on sequence labeling with recurrent neural networks (Klerke et al., 2016; Søgaard and Goldberg, 2016; Bollman and Søgaard, 2016; Plank, 2016; Braud et al., 2016; Martínez Alonso and Plank, 2017), although sequence-to-sequence models have been shown to profit from MTL as well (Luong et al., 2016). Our multi-task learning architecture is similar to the former, with a bi-directional LSTM as a single hidden layer of 100 dimensions that is shared across all tasks. The inputs to this hidden layer are 100-dimensional word vectors that are initialized with pretrained GloVe embeddings, but updated during training. The embedding parameters are also shared. The model then generates predictions from the bi-LSTM through task-specific dense projections. Our model is symmetric in the sense that it does not distinguish between main and auxiliary tasks.

In our MTL setup, a training step consists of uniformly drawing a training task, then sampling a random batch of 32 examples from the task's training data. Every training step thus works on exactly one task, and optimizes the task-specific projection and the shared parameters using Adadelta. As already mentioned, we keep hyper-parameters fixed across single-task and multi-task settings, making our results only applicable to the scenario where one wants to know whether MTL works in the current parameter setting (Collobert and Weston, 2008; Klerke et al., 2016; Søgaard and Goldberg, 2016; Bollman and Søgaard, 2016; Plank, 2016; Braud et al., 2016; Martínez Alonso and Plank, 2017).

3.2 Tasks

In our experiments below, we consider the following ten NLP tasks, with one dataset for each task. Characteristics of the datasets that we use are summarized in Table 1.

1. **CCG Tagging** (CCG) is a sequence tagging problem that assigns a logical type to every token. We use the standard splits for CCG super-tagging from the CCGBank (Hockenmaier and Steedman, 2007).

2. **Chunking** (CHU) identifies continuous spans of tokens that form syntactic units such as noun phrases or verb phrases. We use the standard splits for syntactic chunking from the English Penn Treebank (Marcus et al., 1993).

3. **Sentence Compression** (COM) We use the publicly available subset of the Google Compression dataset (Filippova and Altun, 2013), which has token-level annotations of word deletions.

Task	Size	# Labels	Tok/typ	%OOV	$H(y)$	$\|\|X\|\|_F$	JSD	F_1
CCG	39,604	1,285	23.08	1.13	3.28	981.3	0.41	86.1
CHU	8,936	22	12.01	1.35	1.84	466.4	0.47	93.9
COM	9,600	2	9.47	0.99	0.47	519.3	0.44	51.9
FNT	3,711	2	8.44	1.79	0.51	286.8	0.30	58.0
POS	1,002	12	3.24	14.15	2.27	116.9	0.24	82.6
HYP	2,000	2	6.14	2.14	0.47	269.3	0.48	39.3
KEY	2,398	2	9.10	4.46	0.61	289.1	0.39	64.5
MWE	3,312	3	9.07	0.73	0.53	217.3	0.18	43.3
SEM	15,465	73	11.16	4.72	2.19	614.6	0.35	70.8
STR	3,312	118	9.07	0.73	2.43	217.3	0.26	61.5

Table 1: Dataset characteristics for the individual tasks as defined in Table 2, as well as single-task model performance on test data (micro-averaged F_1).

4. **Semantic frames** (FNT) We use FrameNet 1.5 for jointly predicting target words that trigger frames, and deciding on the correct frame in context.

5. **POS tagging** (POS) We use a dataset of tweets annotated for Universal part-of-speech tags (Petrov et al., 2011).

6. **Hyperlink Prediction** (HYP) We use the hypertext corpus from Spitkovsky et al. (2010) and predict what sequences of words have been bracketed with hyperlinks.

7. **Keyphrase Detection** (KEY) This task amounts to detecting keyphrases in scientific publications. We use the SemEval 2017 Task 10 dataset.

8. **MWE Detection** (MWE) We use the Streusle corpus (Schneider and Smith, 2015) to learn to identify multi-word expressions (*on my own, cope with*).

9. **Super-sense tagging** (SEM) We use the standard splits for the Semcor dataset, predicting coarse-grained semantic types of nouns and verbs (super-senses).

10. **Super-sense Tagging** (STR) As for the MWE task, we use the Streusle corpus, jointly predicting brackets and coarse-grained semantic types of the multi-word expressions.

Data features	
Size	Number of training sentences.
# Labels	The number of labels.
Tokens/types	Type/token ratio in training data.
OOV rate	Percentage of training words not in GloVe vectors.
Label Entropy	Entropy of the label distribution.
Frobenius norm	$\|\|X\|\|_F = [\sum_{i,j} X_{i,j}^2]^{1/2}$, where $X_{i,j}$ is the frequency of term j in sentence i.
JSD	Jensen-Shannon Divergence between train and test bags-of-words.
Learning curve features	
Curve gradients	See text.
Fitted log-curve	See text.

Table 2: Task features

4 Experiments

We train single-task bi-LSTMs for each of the ten tasks, as well as one multi-task model for each of the pairs between the tasks, yielding 90 directed pairs of the form $\langle \mathcal{T}_{main}, \{\mathcal{T}_{main}, \mathcal{T}_{aux}\}\rangle$. The single-task models are trained for 25,000 batches, while multi-task models are trained for 50,000 batches to account for the uniform drawing of the two tasks at every iteration in the multi-task setup. The relative gains and losses from MTL over the single-task models (see Table 1) are presented in Figure 1, showing improvements in 40 out of 90 cases. We see that chunking and high-level semantic tagging generally contribute most to other tasks, while hyperlinks do not significantly improve any other task. On the receiving end, we see that multiword and hyperlink detection seem to profit most from several auxiliary tasks. Symbiotic relationships are formed, e.g., by POS and CCG-tagging, or MWE and compression.

We now investigate whether we can predict gains from MTL given features of the tasks and single-task learning characteristics. We will use

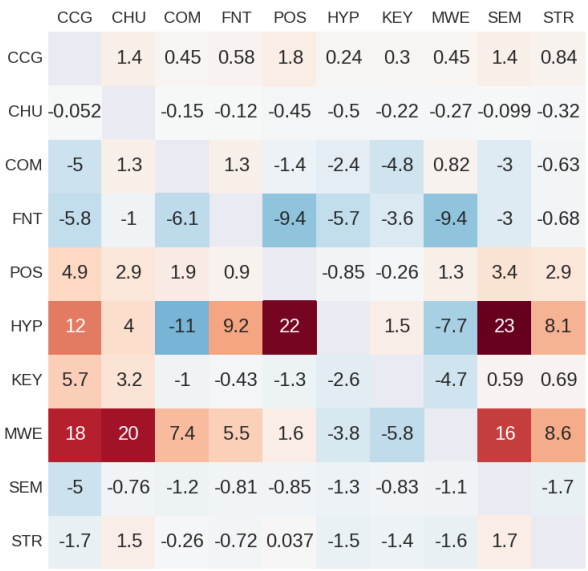

Figure 1: Relative gains and losses (in percent) over main task micro-averaged F_1 when incorporating auxiliary tasks (columns) compared to single-task models for the main tasks (rows).

the induced meta-learning for analyzing what such characteristics are predictive of gains.

Specifically, for each task considered, we extract a number of dataset-inherent features (see Table 2) as well as features that we derive from the learning curve of the respective single-task model. For the curve gradients, we compute the gradients of the loss curve at 10, 20, 30, 50 and 70 percent of the 25,000 batches. For the fitted log-curve parameters, we fit a logarithmic function to the loss curve values, where the function is of the form: $L(i) = a \cdot \ln(c \cdot i + d) + b$. We include the fitted parameters a and c as features that describe the steepness of the learning curve. In total, both the main and the auxiliary task are described by 14 features. Since we also compute the main/auxiliary ratios of these values, each of our 90 data points is described by 42 features that we normalize to the $[0, 1]$ interval. We binarize the results presented in Figure 1 and use logistic regression to predict benefits or detriments of MTL setups based on the features computed above.[1]

4.1 Results

The mean performance of 100 runs of randomized five-fold cross-validation of our logistic regression model for different feature combinations is listed in Table 3. The first observation is that there is a strong signal in our meta-learning features. In almost four in five cases, we can predict the outcome of the MTL experiment from the data and the single task experiments, which gives validity to our feature analysis. We also see that the features derived from the single task inductions are the most important. In fact, using only data-inherent features, the F_1 score of the positive class is worse than the majority baseline.

	Acc.	F_1 (gain)
Majority baseline	0.555	0.615
All features	0.749	0.669
Best, data features only	0.665	0.542
Best combination	0.785	0.713

Table 3: Mean performance across 100 runs of 5-fold CV logistic regression.

4.2 Analysis

Table 4 lists the coefficients for all 42 features. We find that features describing the learning curves for the main and auxiliary tasks are the best predictors of MTL gains. The ratios of the learning curve features seem less predictive, and the gradients around 20-30% seem most important, after the area where the curve typically flattens a bit (around 10%). Interestingly, however, these gradients correlate in opposite ways for the main and auxiliary tasks. The pattern is that if the main tasks have flattening learning curves (small negative gradients) in the 20-30% percentile, but the auxiliary task curves are still relatively steep, MTL is more likely to work. In other words, *multi-task gains are more likely for target tasks that quickly plateau with non-plateauing auxiliary tasks*. We speculate the reason for this is that multi-task learning can help target tasks that get stuck early in local minima, especially if the auxiliary task does not always get stuck fast.

Other features that are predictive include the number of labels in the main task, as well as the label entropy of the auxiliary task. The latter supports the hypothesis put forward by Martínez Alonso and Plank (2017) (see Related work). Note, however, that this may be a side effect of tasks with more uniform label distributions being easier to learn. The out-of-vocabulary rate for the target task also was predictive, which

[1] An experiment in which we tried to predict the magnitude of the losses and gains with linear regression yielded inconclusive results.

Feature	Task	Coefficient
Curve grad. (30%)	Main	-1.566
Curve grad. (20%)	Main	-1.164
Curve param. c	Main	1.007
# Labels	Main	0.828
Label Entropy	Aux	0.798
Curve grad. (30%)	Aux	0.791
Curve grad. (50%)	Main	0.781
OOV rate	Main	0.697
OOV rate	Main/Aux	0.678
Curve grad. (20%)	Aux	0.575
Fr. norm	Main	-0.516
# Labels	Main/Aux	0.504
Curve grad. (70%)	Main	0.434
Label entropy	Main/Aux	-0.411
Fr. norm	Aux	0.346
Tokens/types	Main	-0.297
Curve param. a	Aux	-0.297
Curve grad. (70%)	Aux	-0.279
Curve grad. (10%)	Aux	0.267
Tokens/types	Aux	0.254
Curve param. a	Main/Aux	-0.241
Size	Aux	0.237
Fr. norm	Main/Aux	-0.233
JSD	Aux	-0.207
# Labels	Aux	-0.184
Curve param. c	Aux	-0.174
Tokens/types	Main/Aux	-0.117
Curve param. c	Main/Aux	-0.104
Curve grad. (20%)	Main/Aux	0.104
Label entropy	Main	-0.102
Curve grad. (50%)	Aux	-0.099
Curve grad. (50%)	Main/Aux	0.076
OOV rate	Aux	0.061
Curve grad. (30%)	Main/Aux	-0.060
Size	Main	-0.032
Curve param. a	Main	0.027
Curve grad. (10%)	Main/Aux	0.023
JSD	Main	0.019
JSD	Main/Aux	-0.015
Curve grad. (10%)	Main	$6 \cdot 10^{-2}$
Size	Main/Aux	$-6 \cdot 10^{-3}$
Curve grad. (70%)	Main/Aux	$-4 \cdot 10^{-4}$

Table 4: Predictors of MTL benefit by logistic regression model coefficient (absolute value).

makes sense as the embedding parameters are also updated when learning from the auxiliary data.

Less predictive features include Jensen-Shannon divergences, which is surprising, since multi-task learning is often treated as a transfer learning algorithm (Søgaard and Goldberg, 2016). It is also surprising to see that size differences between the datasets are not very predictive.

5 Conclusion and Future Work

We present the first systematic study of when MTL works in the context of common NLP tasks, when single task parameter settings are also applied for multi-task learning. Key findings include that MTL gains are predictable from dataset characteristics and features extracted from the single-task inductions. We also show that the most predictive features relate to the single-task learning curves, suggesting that MTL, when successful, often helps target tasks out of local minima. We also observed that label entropy in the auxiliary task was also a good predictor, lending some support to the hypothesis in Martínez Alonso and Plank (2017); but there was little evidence that dataset balance is a reliable predictor, unlike what previous work has suggested.

In future work, we aim to extend our experiments to a setting where we optimize hyperparameters for the single- and multi-task models individually, which will give us a more reliable picture of the effect to be expected from multi-task learning in the wild. Generally, further conclusions could be drawn from settings where the joint models do not treat the two tasks as equals, but instead give more importance to the main task, for instance through a non-uniform drawing of the task considered at each training iteration, or through an adaptation of the learning rates. We are also interested in extending this work to additional NLP tasks, including tasks that go beyond sequence labeling such as language modeling or sequence-to-sequence problems.

Acknowledgments

For valuable comments, we would like to thank Dirk Hovy, Yoav Goldberg, the attendants at the second author's invited talk at the Danish Society for Statistics, as well as the anonymous reviewers. This research was partially funded by the ERC Starting Grant LOWLANDS No. 313695, as well as by Trygfonden.

References

Jonathan Baxter. 2000. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198.

Shai Ben-David and Reba Borberly. 2003. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning*, 73:273–287.

Adrian Benton, Margaret Mitchell, and Dirk Hovy. 2017. Multitask learning for mental health conditions with limited social media data. In *EACL*.

Marcel Bollman and Anders Søgaard. 2016. Improving historical spelling normalization with bi-directional lstms and multi-task learning. In *COLING*.

Chloe Braud, Barbara Plank, and Anders Søgaard. 2016. Multi-view and multi-task training of rst discourse parser. In *COLING*.

Rich Caruana. 1993. Multitask learning: a knowledge-based source of inductive bias. In *ICML*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *EMNLP*, pages 1481–1491.

Julia Hockenmaier and Mark Steedman. 2007. Ccgbank: A corpus of ccg derivations and dependency structures extracted from the penn treebank. *Comput. Linguist.*, 33(3):355–396, September.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016. Improving sentence compression by learning to predict gaze. In *NAACL*.

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

Mitchell Marcus, Mary Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Héctor Martínez Alonso and Barbara Plank. 2017. Multitask learning for semantic sequence prediction under varying data conditions. In *EACL*.

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

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. CoRR abs/1104.2086.

Barbara Plank. 2016. Keystroke dynamics as signal for shallow syntactic parsing. In *COLING*.

Nathan Schneider and Noah A Smith. 2015. A corpus and model integrating multiword expressions and supersenses. *Proc. of NAACL-HLT. Denver, Colorado, USA*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multitask learning with low level tasks supervised at lower layers. In *ACL*.

Valentin I Spitkovsky, Daniel Jurafsky, and Hiyan Alshawi. 2010. Profiting from mark-up: Hyper-text annotations for guided parsing. In *ACL*.