# Multi-Task Learning for Sequence Tagging: An Empirical Study

**Soravit Changpinyo, Hexiang Hu,** and **Fei Sha**
Department of Computer Science
University of Southern California
Los Angeles, CA 90089
schangpi,hexiangh,feisha@usc.edu

## Abstract

We study three general multi-task learning (MTL) approaches on 11 sequence tagging tasks. Our extensive empirical results show that in about 50% of the cases, jointly learning all 11 tasks improves upon either independent or pairwise learning of the tasks. We also show that pairwise MTL can inform us what tasks can benefit others or what tasks can be benefited if they are learned jointly. In particular, we identify tasks that can always benefit others as well as tasks that can always be harmed by others. Interestingly, one of our MTL approaches yields embeddings of the tasks that reveal the natural clustering of semantic and syntactic tasks. Our inquiries have opened the doors to further utilization of MTL in NLP.

## 1 Introduction

Multi-task learning (MTL) has long been studied in the machine learning literature, cf. (Caruana, 1997). The technique has also been popular in NLP, for example, in (Collobert and Weston, 2008; Collobert et al., 2011; Luong et al., 2016). The main thesis underpinning MTL is that solving many tasks together provides a shared inductive bias that leads to more robust and generalizable systems. This is especially appealing for NLP as data for many tasks are scarce — shared learning thus reduces the amount of training data needed. MTL has been validated in recent work, mostly where auxiliary tasks are used to improve the performance on a target task, for example, in sequence tagging (Søgaard and Goldberg, 2016; Bjerva et al., 2016; Plank et al., 2016; Alonso and Plank, 2017; Bingel and Søgaard, 2017).

Despite those successful applications, several key issues about the effectiveness of MTL remain open. Firstly, with only a few exceptions, much existing work focuses on "pairwise" MTL where there is a target task and one or several (carefully) selected auxiliary tasks. However, *can jointly learning many tasks benefit all of them together*? A positive answer will significantly raise the utility of MTL. Secondly, *how are tasks related such that one could benefit another?* For instance, one plausible intuition is that syntactic and semantic tasks might benefit among their two separate groups though cross-group assistance is weak or unlikely. However, such notions have not been put to test thoroughly on a significant number of tasks.

In this paper, we address such questions. We investigate learning jointly multiple sequence tagging tasks. Besides using independent single-task learning as a baseline and a popular shared-encoder MTL framework for sequence tagging (Collobert et al., 2011), we propose two variants of MTL, where both the encoder and the decoder could be shared by all tasks.

We conduct extensive empirical studies on 11 sequence tagging tasks — we defer the discussion on why we select such tasks to a later section. We demonstrate that there is a benefit to moving beyond "pairwise" MTL. We also obtain interesting pairwise relationships that reveal which tasks are beneficial or harmful to others, and which tasks are likely to be benefited or harmed. We find such information correlated with the results of MTL using more than two tasks. We also study selecting only benefiting tasks for joint training, showing that such a "greedy" approach in general improves the MTL performance, highlighting the need of identifying with whom to jointly learn.





*Proceedings of the 27th International Conference on Computational Linguistics*, pages 2965–2977
Santa Fe, New Mexico, USA, August 20-26, 2018.

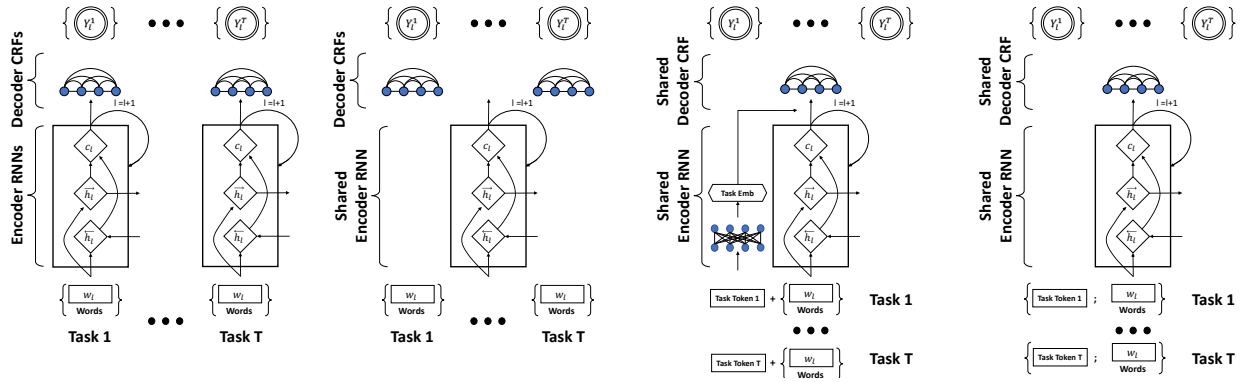
The rest of the paper is organized as follows. We describe different approaches for learning from multiple tasks in Sect. 2. We describe our experimental setup and results in Sect. 3 and Sect. 4, respectively. We discuss related work in Sect. 5. Finally, we conclude with discussion and future work in Sect. 6.

## 2 Multi-Task Learning for Sequence Tagging

In this section, we describe general approaches to multi-task learning (MTL) for sequence tagging. We select sequence tagging tasks for several reasons. Firstly, we want to concentrate on comparing the tasks themselves without being confounded by designing specialized MTL methods for solving complicated tasks. Sequence tagging tasks are done at the word level, allowing us to focus on simpler models while still enabling varying degrees of sharing among tasks. Secondly, those tasks are often the first steps in NLP pipelines that come with extremely diverse resources. Understanding the nature of the relationships between them is likely to have a broad impact on many downstream applications.

Let $\mathsf{T}$ be the number of tasks and $\boldsymbol{D}^t$ be training data of task $t \in \{1, \ldots, \mathsf{T}\}$. A dataset for each task consists of input-output pairs. In sequence tagging, each pair corresponds to a sequence of words $w_{1:\mathsf{L}}$ and their corresponding ground-truth tags $y_{1:\mathsf{L}}$, where $\mathsf{L}$ is the sequence length. We note that our definition of "task" is not the same as "domain" or "dataset." In particular, we differentiate between tasks based on whether or not they share the label space of tags. For instance, part-of-speech tagging on weblog and that on email domains are considered the same task in this paper.

Given the training data $\{\boldsymbol{D}^1, \ldots, \boldsymbol{D}^{\mathsf{T}}\}$, we describe how to learn one or more models to perform all the $\mathsf{T}$ tasks. In general, our models follow the design of state-of-the-art sequence taggers (Reimers and Gurevych, 2017). They have an encoder $e$ with parameters $\theta$ that encodes a sequence of word tokens into a sequence of vectors and a decoder $d$ with parameters $\phi$ that decodes the sequence of vectors into a sequence of predicted tags $\hat{y}_{1:\mathsf{L}}$. That is, $c_{1:\mathsf{L}} = e(w_{1:\mathsf{L}}; \theta)$ and $\hat{y}_{1:\mathsf{L}} = d(c_{1:\mathsf{L}}; \phi)$. The model parameters are learned by minimizing some loss function $\mathcal{L}(\hat{y}_{1:\mathsf{L}}, y_{1:\mathsf{L}})$ over $\theta$ and $\phi$. In what follows, we will use superscripts to differentiate instances from different tasks.




Single-task learning | MTL (**MULTI-DEC**) | MTL (**TE⊕DEC**) | MTL (**TE⊕ENC**)

Figure 1: Different settings for learning from multiple tasks considered in our experiments

In **single-task learning (STL)**, we learn $\mathsf{T}$ models independently. For each task $t$, we have an encoder $e^t(\cdot; \theta^t)$ and a decoder $d^t(\cdot; \phi^t)$. Clearly, information is not shared between tasks in this case.

In **multi-task learning (MTL)**, we consider two or more tasks and train an MTL model *jointly* over a combined loss $\sum_t \mathcal{L}(\hat{y}^t_{1:\mathsf{L}}, y^t_{1:\mathsf{L}})$. In this paper, we consider the following general frameworks that are different in the nature of how the parameters of those tasks are shared.

**Multi-task learning with multiple decoders (Multi-Dec)** We learn a shared encoder $e(\cdot; \theta)$ and $\mathsf{T}$ decoders $\{d^t(\cdot; \theta^t)\}_{t=1}^{\mathsf{T}}$. This setting has been explored for sequence tagging in (Collobert and Weston, 2008; Collobert et al., 2011). In the context of sequence-to-sequence learning (Sutskever et al., 2014), this is similar to the "one-to-many" MTL setting in (Luong et al., 2016).

**Multi-task learning with task embeddings (TE)** We learn a shared encoder $e(\cdot; \theta)$ for the input sentence as well as a shared decoder $d(\cdot; \phi)$. To equip our model with the ability to perform one-to-many mapping (i.e., multiple tasks), we augment the model with "task embeddings." Specifically, we additionally learn

| Dataset | # sentences | Token/type | Task | # labels | Label entropy |
|---|---|---|---|---|---|
| Universal Dependencies v1.4 | 12543/16622 | 12.3/13.2 | UPOS<br>XPOS | 17<br>50 | 2.5<br>3.1 |
| CoNLL-2000 | 8936/10948 | 12.3/13.3 | CHUNK | 42 | 2.3 |
| CoNLL-2003 | 14041/20744 | 9.7/11.2 | NER | 17 | 0.9 |
| Streusle 4.0 | 2723/3812 | 8.6/9.3 | MWE<br>SUPSENSE | 3<br>212 | 0.5<br>2.2 |
| SemCor | 13851/20092 | 13.2/16.2 | SEM<br>SEMTR | 75<br>11 | 2.2<br>1.3 |
| Broadcast News 1 | 880/1370 | 5.2/6.1 | COM | 2 | 0.6 |
| FrameNet 1.5 | 3711/5711 | 8.6/9.1 | FRAME | 2 | 0.5 |
| Hyper-Text Corpus | 2000/3974 | 6.7/9.0 | HYP | 2 | 0.4 |

Table 1: Datasets used in our experiments, as well as their key characteristics and their corresponding tasks. / is used to separate statistics for training data only and those for all subsets of data.

a function $f(t)$ that maps a task ID $t$ to a vector. We explore two ways of injecting task embeddings into models. In both cases, our $f$ is simply an embedding layer that maps the task ID to a dense vector.

One approach, denoted by **TE⊕DEC**, is to incorporate task embeddings into the decoder. We concatenate the task embeddings with the encoder's outputs $c^t_{1:\mathsf{L}}$ and then feed the result to the decoder.

The other approach, denoted by **TE⊕ENC**, is to combine the task embeddings with the input sequence of words at the encoder. We implement this by prepending the task token (`<<upos>>`, `<<chunk>>`, `<<mwe>>`, etc.) to the input sequence and treat the task token as a word token (Johnson et al., 2017).

While the encoder in **TE⊕DEC** must learn to encode a general-purpose representation of the input sentence, the encoder in **TE⊕ENC** knows from the start which task it is going to perform.

Fig. 1 illustrates different settings described above. Clearly, the number of model parameters is reduced significantly when we move from STL to MTL. Which MTL model is more economical depends on several factors, including the number of tasks, the dimension of the encoder output, the general architecture of the decoder, the dimension of task embeddings, how to augment the system with task embeddings, and the degree of tagset overlap.

## 3 Experimental Setup

### 3.1 Datasets and Tasks

Table 1 summarizes the datasets used in our experiments, along with their corresponding tasks and important statistics. Table 2 shows an example of each task's input-output pairs. We describe details below. For all tasks, we use the standard splits unless specified otherwise.

We perform universal and English-specific POS tagging (UPOS and XPOS) on sentences from the English Web Treebank (Bies et al., 2012), annotated by the Universal Dependencies project (Nivre et al., 2016). We perform syntactic chunking (CHUNK) on sentences from the WSJ portion of the Penn Treebank (Marcus et al., 1993), annotated by the CoNLL-2000 shared task (Tjong Kim Sang and Buchholz, 2000). We use sections 15-18 for training. The shared task uses section 20 for testing and does not designate the development set, so we use the first 1001 sentences for development and the rest 1011 for testing. We perform named entity recognition (NER) on sentences from the Reuters Corpus (Lewis et al., 2004), consisting of news stories between August 1996-97, annotated by the CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003). For both CHUNK and NER, we use the IOBES tagging scheme.

We perform multi-word expression identification (MWE) and supersense tagging (SUPSENSE) on sentences from the reviews section of the English Web Treebank, annotated under the Streusle project (Schneider and Smith, 2015)[1]. We perform supersense (SEM) and semantic trait (SEMTR) tagging on SemCor's sentences (Landes et al., 1998), taken from a subset of the Brown Corpus (Francis and Kučera, 1982), using the splits provided by (Alonso and Plank, 2017) for both tasks[2]. For SEM, they are annotated with supersense tags (Miller et al., 1993) by (Ciaramita and Altun, 2006)[3]. For SEMTR, (Alonso and Plank, 2017) uses the EuroWordNet list of ontological types for senses (Vossen et al., 1998) to convert supersenses into coarser semantic traits.

[1] https://github.com/nert-gu/streusle

[2] https://github.com/bplank/multitasksemantics

[3] We consider SUPSENSE and SEM as different tasks as they use different sets of supersense tags.

| Task | Input/Output |
|---|---|
| UPOS | once again , thank you all for an outstanding accomplishment .<br>ADV ADV PUNCT VERB PRON DET ADP DET ADJ NOUN PUNCT |
| XPOS | once again , thank you all for an outstanding accomplishment .<br>RB RB , VBP PRP DT IN DT JJ NN . |
| CHUNK | the carrier also seemed eager to place blame on its american counterparts .<br>B-NP E-NP S-ADVP S-VP S-ADJP B-VP E-VP S-NP S-PP B-NP I-NP E-NP O |
| NER | 6. pier francesco chili ( italy ) ducati 17541<br>O B-PER I-PER E-PER O S-LOC O S-ORG O |
| MWE | had to keep in mind that the a / c broke , i feel bad it was their opening !<br>B I B I I O O B I I O O O O O O O O O |
| SUPSENSE | this place may have been something sometime ; but it way past it " sell by date " .<br>O n.GROUP O O v.stative O O O O O O p.Time p.Gestalt O v.possession p.Time n.TIME O O |
| SEM | a hypothetical example will illustrate this point .<br>O adj.all noun.cognition O verb.communication O noun.communication O |
| SEMTR | he wondered if the audience would let him finish .<br>O Mental O O Object O Agentive O BoundedEvent O |
| COM | he made the decisions in 1995 , in early 1996 , to spend at a very high rate .<br>KEEP KEEP DEL KEEP DEL DEL DEL DEL DEL DEL DEL KEEP KEEP KEEP KEEP DEL KEEP KEEP KEEP |
| FRAME | please continue our important partnership .<br>O B-TARGET O B-TARGET O O |
| HYP | will this incident lead to a further separation of civilizations ?<br>O O O O O O O B-HTML B-HTML B-HTML O |

Table 2: Examples of input-output pairs of the tasks in consideration

For sentence compression (COM), we identify which words to keep in a compressed version of sentences from the 1996 English Broadcast News Speech (HUB4) (Graff, 1997), created by (Clarke and Lapata, 2006)[4]. We use the labels from the first annotator. For frame target identification (FRAME), we detect words that evoke frames (Das et al., 2014) on sentences from the British National Corpus, annotated under the FrameNet project (Baker et al., 1998). For both COM and FRAME, we use the splits provided by (Bingel and Søgaard, 2017). For hyper-link detection (HYP), we identify which words in the sequence are marked with hyperlinks on text from Daniel Pipes' news-style blog collected by (Spitkovsky et al., 2010)[5]. We use the "select" subset that correspond to marked, complete sentences.

### 3.2 Metrics and Score Comparison

We use the span-based micro-averaged F1 score (without the O tag) for all tasks. We run each configuration three times with different initializations and compute mean and standard deviation of the scores. To compare two scores, we use the following strategy. Let $\mu_1$, $\sigma_1$ and $\mu_2$, $\sigma_2$ be two sets of scores (mean and std, respectively). We say that $\mu_1$ is "higher" than $\mu_2$ if $\mu_1 - k \times \sigma_1 > \mu_2 + k \times \sigma_2$, where $k$ is a parameter that controls how strict we want the definition to be. "lower" is defined in the same manner with $>$ changed to $<$ and $-$ switched with $+$. $k$ is set to 1.5 in all of our experiments.

### 3.3 Models

**General architectures** We use bidirectional recurrent neural networks (biRNNs) as our encoders for both words and characters (Irsoy and Cardie, 2014; Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016). Our word/character sequence encoders and decoder classifiers are common in literature and most similar to (Lample et al., 2016), but we use two-layer biRNNs (instead of one) with Gated Recurrent Unit (GRU) (Cho et al., 2014) (instead of with LSTM (Hochreiter and Schmidhuber, 1997)).

Each word is represented by a 100-dimensional vector that is the concatenation of a 50-dimensional *embedding* vector and the 50-dimensional output of a *character* biRNN (whose hidden representation dimension is 25 in each direction). We feed a sequence of those 100-dimensional representations to a *word* biRNN, whose hidden representation dimension is 300 in each direction, resulting in a sequence of 600-dimensional vectors. In **TE⊕DEC**, the token encoder is also used to encode a task token (which is then concatenated to the encoder's output), where each task is represented as a 25-dimensional vector. For decoder/classifiers, we predict a sequence of tags using a linear projection layer (to the tagset size) followed by a conditional random field (CRF) (Lafferty et al., 2001).

**Implementation and training details** We implement our models in PyTorch (Paszke et al., 2017) on top of the AllenNLP library (Gardner et al., 2018). Code is to be available at https://github.

[4] http://jamesclarke.net/research/resources/
[5] https://nlp.stanford.edu/valentin/pubs/markup-data.tar.bz2

com/schangpi/.

Words are lower-cased, but characters are not. Word embeddings are initialized with GloVe (Pennington et al., 2014) trained on Wikipedia 2014 and Gigaword 5. We use strategies suggested by (Ma and Hovy, 2016) for initializing other parameters in our networks. Character embeddings are initialized uniformly in $[-\sqrt{3/d}, \sqrt{3/d}]$, where $d$ is the dimension of the embeddings. Weight matrices are initialized with Xavier Uniform (Glorot and Bengio, 2010), i.e., uniformly in $[-\sqrt{6/(r+c)}, \sqrt{6/(r+c)}]$, where $r$ and $c$ are the number of of rows and columns in the structure. Bias vectors are initialized with zeros.

We use Adam (Kingma and Ba, 2015) with default hyperparameters and a mini-batch size of 32. The dropout rate is 0.25 for the character encoder and 0.5 for the word encoder. We use gradient normalization (Pascanu et al., 2013) with a threshold of 5. We halve the learning rate if the validation performance does not improve for two epochs, and stop training if the validation performance does not improve for 10 epochs. We use L2 regularization with parameter 0.01 for the transition matrix of the CRF.

For the training of MTL models, we make sure that each mini-batch is balanced; the difference in numbers of examples from any pair of tasks is no more than 1. As a result, each epoch may not go through all examples of some tasks whose training set sizes are large. In a similar manner, during validation, the average F1 score is over all tasks rather than over all validation examples.

### 3.4 Various Settings for Learning from Multiple Tasks

We consider the following settings: (i) "STL" where we train each model on one task alone; (ii) "Pairwise MTL" where we train on two tasks jointly; (iii) "All MTL" where we train on all tasks jointly; (iv) "Oracle MTL" where we train on the Oracle set of the testing task jointly with the testing task; (v) "All-but-one MTL" setting where we train on all tasks jointly except for one (as part of Sect. 4.4.)

**Constructing the Oracle Set of a Testing Task** The Oracle set of a task $t$ is constructed from the pairwise performances: let $\mu(A,t), \sigma(A,t)$ be the F1 score and the standard deviation of a model that is jointly trained on a set of tasks in the set $A$ and that is tested on task $t$. Task $s$ is considered "beneficial" to another (testing) task $t$ if $\mu(\{s,t\},t)$ is "higher" than $\mu(\{t\},t)$ (cf. Sect. 3.2). Then, the "Oracle" set for a task $t$ is the set of its all beneficial (single) tasks. Throughout our experiments, we compute $\mu$ and $\sigma$ by averaging over three rounds (cf. Sect. 3.2, standard deviations can be found on the arXiv version.)

## 4 Results and Analysis

### 4.1 Main Results

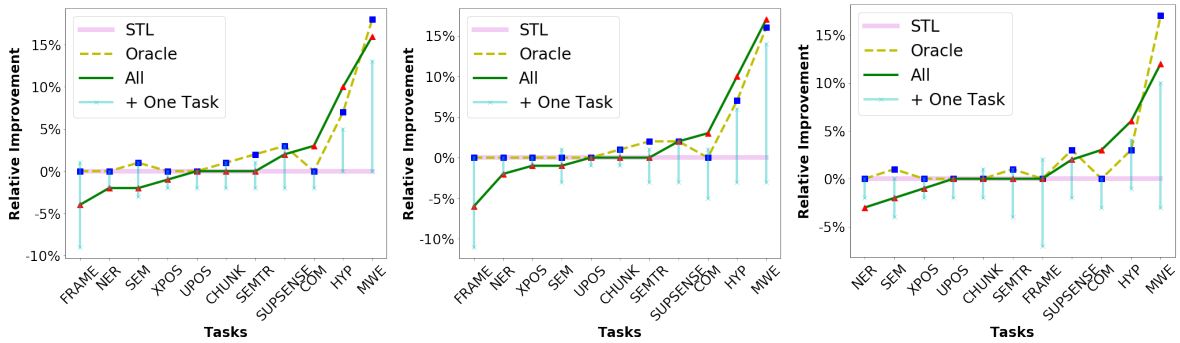



Figure 2: Summary of our results for MTL methods **MULTI-DEC** (left), **TE⊕DEC** (middle), and **TE⊕ENC** (right) on various settings with different types of sharing. The vertical axis is the relative improvement over STL. See texts for details. Best viewed in color.

Fig. 2 summarizes our main findings. We compare relative improvement over single-task learning (STL) between various settings with different types of sharing in Sect. 3.4. Scores from the pairwise setting ("+One Task") are represented as a vertical bar, delineating the maximum and minimum improvement over STL by jointly learning a task with one of the remaining 10 tasks. The "All" setting (red triangles) indicates the joint learning all 11 tasks. The "Oracle" setting (blue rectangles) indicates the joint learning using a subset of 11 tasks which are deemed beneficial, based on corresponding performances in pairwise MTL, as defined in Sect. 3.4.

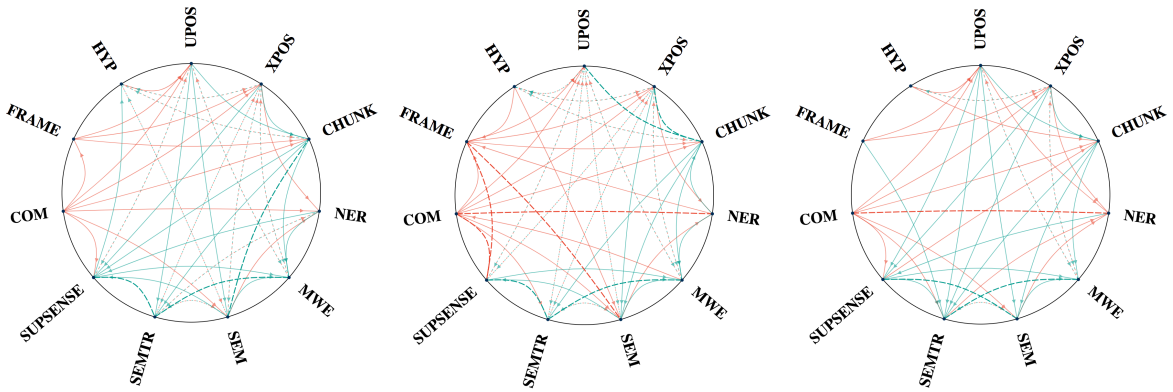



Figure 3: Pairwise MTL relationships (benefit vs. harm) using **MULTI-DEC** (left), **TE⊕DEC** (middle), and **TE⊕ENC** (right). **Solid** green (red) directed edge from $s$ to $t$ denotes $s$ benefiting (harming) $t$. **Dashed** Green (Red) edges between $s$ and $t$ denote they benefiting (harming) *each other*. **Dotted** edges denote asymmetric relationship: benefit in one direction but harm in the reverse direction. Absence of edges denotes neutral relationships. *Best viewed in color and with a zoom-in.*

We observe that (1) [STL vs. Pairwise/All] Neither pairwise MTL nor All always improves upon STL; (2) [STL vs. Oracle] Oracle in general outperforms or at least does not worsen STL; (3) [All/Oracle vs. Pairwise] All does better than Pairwise on about half of the cases, while Oracle almost always does better than Pairwise; (4) [All vs. Oracle] Consider when both All and Oracle improve upon STL. For **MULTI-DEC** and **TE⊕ENC**, Oracle generally dominates All, except on the task HYP. For **TE⊕DEC**, their magnitudes of improvement are mostly comparable, except on SEMTR (Oracle is better) and on HYP (All is better). In addition, All is better than Oracle on the task COM, in which case Oracle is STL.

In the arXiv version, we compare different MTL approaches: **MULTI-DEC**, **TE⊕DEC**, and **TE⊕ENC**. There is no significant difference among them.

### 4.2 Pairwise MTL results

**Summary** The summary plot in Fig. 3 gives a bird's-eye view of patterns in which a task might benefit or harm another one. For example, MWE is always benefited from jointly learning any of the 10 tasks as the *incoming edges* are green, so is SEMTR in most cases. On the other end, COM seems to be harming any of the 10 as the *outgoing edges* are almost always red. For CHUNK and U/XPOS, it generally benefits others (or at least does not do harm) as most of their *outgoing* edges are green.

In Table 3-5, we report F1 scores for **MULTI-DEC**, **TE⊕DEC**, and **TE⊕ENC**, respectively. In each table, rows denote settings in which we train our models, and columns correspond to tasks we test them on. We also include "Average" of all pairwise scores, as well as the number of positive (↑) and negative (↓) relationships in each row or each column.

**Which tasks are benefited or harmed by others in pairwise MTL?** MWE, SUPSENSE, SEMTR, and HYP are generally benefited by other tasks. The improvement is more significant in MWE and HYP. UPOS, XPOS, NER, COM, and FRAME (**MULTI-DEC** and **TE⊕DEC**) are often hurt by other tasks. Finally, the results are mixed for CHUNK and SEM.

**Which tasks are beneficial or harmful?** UPOS, XPOS, and CHUNK are universal helpers, beneficial in 16, 17, and 14 cases, while harmful only in 1, 3, and 0 cases, respectively. Interestingly, CHUNK never hurts any task, while both UPOS and XPOS can be harmful to NER. While CHUNK is considered more of a syntactic task, the fact that it informs other tasks about the boundaries of phrases may aid the learning of other semantic tasks (task embeddings in Sect. 4.4 seem to support this hypothesis).

On the other hand, COM, FRAME, and HYP are generally harmful, all useful in 0 cases and causing the performance drop in 22, 10, 12 cases, respectively. One factor that may play a role is the training set sizes of these tasks. However, we note that both MWE and SUPSENSE (Streusle dataset) has smaller training set sizes than FRAME does, but those tasks can still benefit some tasks. (On the other hand, NER has the largest training set, but infrequently benefits other tasks, less frequently than SUPSENSE does.) Another potential cause is the fact that all those harmful tasks have the smallest label size of 2. This combined with small dataset sizes leads to a higher chance of overfitting. Finally, it may be possible that harmful

tasks are simply unrelated; for example, the nature of COM, FRAME, or HYP may be very different from other tasks — an entirely different kind of reasoning is required.

Finally, NER, MWE, SEM, SEMTR, and SUPSENSE can be beneficial or harmful, depending on which other tasks they are trained with.

| | UPOS | XPOS | CHUNK | NER | MWE | SEM | SEMTR | SUPSENSE | COM | FRAME | HYP | #↑ | #↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +UPOS | 95.4 | 95.01 | 94.18 ↑ | 87.68 | 59.99 ↑ | 73.23 ↑ | 74.93 ↑ | 68.25 ↑ | 72.46 | 62.14 | 48.02 | 5 | 0 |
| +XPOS | 95.38 | 95.04 | 93.97 ↑ | 87.61 ↓ | 58.87 ↑ | 73.34 ↑ | 74.91 ↑ | 67.78 ↑ | 72.83 | 60.77 | 48.81 ↑ | 6 | 1 |
| +CHUNK | 95.43 | 95.1 | 93.49 | 87.96 | 59.18 ↑ | 73.16 ↑ | 74.79 ↑ | 67.39 ↑ | 72.44 | 62.67 | 47.85 ↑ | 5 | 0 |
| +NER | 95.38 | 94.98 | 93.47 | 88.24 | 55.4 ↑ | 72.88 | 74.34 ↑ | 68.06 ↑ | 70.93 | 62.39 | 47.9 | 3 | 0 |
| +MWE | 95.15 ↓ | 94.7 ↓ | 93.54 | 88.15 | 53.07 | 72.75 | 74.51 ↑ | 66.88 | 71.31 | 61.75 | 47.32 | 1 | 2 |
| +SEM | 95.23 | 94.77 ↓ | 93.63 ↑ | 87.35 ↓ | 60.16 ↑ | 72.77 | 74.73 ↑ | 68.29 ↑ | 72.72 | 61.74 | 48.15 ↑ | 5 | 2 |
| +SEMTR | 95.17 | 94.86 ↓ | 93.61 | 87.34 ↓ | 58.84 ↑ | 72.5 ↓ | 74.02 | 68.6 ↑ | 71.96 | 62.03 | 47.74 | 2 | 3 |
| +SUPSENSE | 95.08 ↓ | 94.75 | 93.2 | 87.9 | 58.81 ↑ | 72.81 | 74.61 ↑ | 66.81 | 72.24 | 61.94 | 49.23 ↑ | 3 | 1 |
| +COM | 93.04 ↓ | 93.19 ↓ | 91.94 ↓ | 86.62 ↓ | 53.89 | 70.39 ↓ | 72.6 | 65.57 ↓ | 72.71 | 56.52 ↓ | 47.41 | 0 | 7 |
| +FRAME | 94.98 ↓ | 94.64 ↓ | 93.22 ↓ | 88.15 | 53.88 | 72.76 | 74.18 | 66.59 | 72.47 | 62.04 | 47.5 | 0 | 3 |
| +HYP | 94.84 ↓ | 94.46 ↓ | 92.96 ↓ | 87.98 | 53.08 | 72.47 ↓ | 74.23 | 66.47 | 71.82 | 61.02 | 46.73 | 0 | 4 |
| #↑ | 0 | 0 | 3 | 0 | 7 | 3 | 7 | 6 | 0 | 0 | 4 | | |
| #↓ | 5 | 6 | 3 | 4 | 0 | 3 | 0 | 1 | 0 | 1 | 0 | | |
| Average | 94.97 | 94.65 | 93.37 | 87.67 | 57.21 | 72.63 | 74.38 | 67.39 | 72.12 | 61.3 | 47.99 | | |
| All | 95.04 ↓ | 94.31 ↓ | 93.44 | 86.38 ↓ | 61.43 ↑ | 71.53 ↓ | 74.26 ↑ | 68.1 ↑ | 74.54 | 59.71 | 51.41 ↑ | 4 | 4 |
| Oracle | 95.4 | 95.04 | 94.01 ↑ | 88.24 | 62.76 ↑ | 73.32 ↑ | 75.23 ↑ | 68.53 ↑ | 72.71 | 62.04 | 50.0 ↑ | 6 | 0 |

Table 3: F1 scores for **MULTI-DEC**. We compare STL setting (blue), with pairwise MTL (+⟨task⟩), All, and Oracle. We test on each task in the columns. Beneficial settings are in green. Harmful setting are in red. The last two columns indicate how many tasks are helped or harmed by the task at that row.

| | UPOS | XPOS | CHUNK | NER | MWE | SEM | SEMTR | SUPSENSE | COM | FRAME | HYP | #↑ | #↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +UPOS | 95.4 | 94.99 | 94.02 ↑ | 87.99 | 60.28 ↑ | 73.17 ↑ | 74.87 ↑ | 67.8 ↑ | 72.86 | 61.54 | 49.36 ↑ | 6 | 0 |
| +XPOS | 95.4 | 95.04 | 94.18 ↑ | 87.65 ↓ | 60.32 ↑ | 73.21 ↑ | 74.84 ↑ | 68.3 ↑ | 72.87 | 61.44 | 49.23 ↑ | 6 | 1 |
| +CHUNK | 95.57 ↑ | 95.21 ↑ | 93.49 | 88.11 | 57.61 ↑ | 73.02 ↑ | 74.73 ↑ | 67.29 | 73.3 | 61.39 | 48.43 ↑ | 6 | 0 |
| +NER | 95.32 | 95.09 | 93.64 ↑ | 88.24 | 55.17 ↑ | 72.77 | 74.01 | 67.25 | 71.08 ↓ | 59.25 ↓ | 48.24 | 2 | 2 |
| +MWE | 95.11 ↓ | 94.8 ↓ | 93.59 | 87.99 | 53.07 | 72.66 | 74.63 ↑ | 66.88 | 70.93 ↓ | 56.77 | 45.83 | 1 | 3 |
| +SEM | 95.2 ↓ | 94.82 | 93.45 | 87.27 ↓ | 58.21 ↑ | 72.77 | 74.72 ↑ | 68.46 ↑ | 73.14 | 60.09 ↓ | 47.95 | 3 | 3 |
| +SEMTR | 95.21 ↓ | 94.8 ↓ | 93.47 | 87.75 | 58.55 ↑ | 72.5 ↓ | 74.02 | 68.18 ↑ | 71.74 | 59.77 | 46.96 | 2 | 3 |
| +SUPSENSE | 95.05 ↓ | 94.81 ↓ | 93.25 | 87.94 | 58.75 ↑ | 72.71 | 74.52 ↑ | 66.81 | 69.13 ↓ | 55.68 ↓ | 47.29 | 2 | 4 |
| +COM | 94.03 ↓ | 93.94 ↓ | 92.29 ↓ | 86.59 ↓ | 51.72 | 70.37 ↓ | 71.76 ↓ | 64.98 ↓ | 72.71 | 55.25 ↓ | 45.24 | 0 | 8 |
| +FRAME | 94.79 ↓ | 94.66 ↓ | 93.23 ↓ | 88.02 | 53.05 | 72.26 ↓ | 74.21 | 66.2 ↓ | 72.89 | 62.04 | 46.0 | 0 | 5 |
| +HYP | 94.35 ↓ | 94.56 ↓ | 92.86 ↓ | 87.91 | 52.98 | 72.15 ↓ | 74.19 | 66.52 | 70.47 | 55.35 ↓ | 46.73 | 0 | 5 |
| #↑ | 1 | 1 | 3 | 0 | 7 | 3 | 6 | 4 | 0 | 0 | 3 | | |
| #↓ | 7 | 6 | 3 | 3 | 0 | 4 | 1 | 2 | 3 | 5 | 0 | | |
| Average | 95.0 | 94.77 | 93.4 | 87.72 | 56.67 | 72.48 | 74.25 | 67.19 | 71.84 | 58.65 | 47.45 | | |
| All | 94.95 ↓ | 94.42 ↓ | 93.64 | 86.8 ↓ | 61.97 ↑ | 71.72 ↓ | 74.36 ↑ | 67.98 ↑ | 74.61 ↑ | 58.14 ↓ | 51.31 ↑ | 5 | 5 |
| Oracle | 95.57 ↑ | 95.21 ↑ | 94.07 ↑ | 88.24 | 61.74 ↑ | 73.1 ↑ | 75.24 ↑ | 68.22 ↑ | 72.71 | 62.04 | 50.15 ↑ | 8 | 0 |

Table 4: F1 scores for **TE⊕DEC**. We compare STL setting (blue), with pairwise MTL (+⟨task⟩), All, and Oracle. We test on each task in the columns. Beneficial settings are in green. Harmful setting are in red. The last two columns indicate how many tasks are helped or harmed by the task at that row.

### 4.3 All MTL Results

In addition to pairwise MTL results, we report the performances in the All and Oracle MTL settings in the last two rows of Table 3-5. We find that their performances depend largely on the trend in their corresponding pairwise MTL. We provide examples and discussion of such observations below.

**How much is STL vs. Pairwise MTL predictive of STL vs. All MTL?** We find that the performance of pairwise MTL is predictive of the performance of All MTL to some degree. Below we discuss the results in more detail. Note that we would like to be predictive in both the performance direction and magnitude (whether and how much the scores will improve or degrade over the baseline).

*When pairwise MTL improves upon STL even slightly, All improves upon STL in all cases* (MWE, SEMTR, SUPSENSE, and HYP). This is despite the fact that jointly learning some pairs of tasks lead to performance degradation (COM and FRAME in the case of SUPSENSE and COM in the case of SEMTR). Furthermore, when pairwise MTL leads to improvement in all cases (all pairwise rows in MWE and HYP), All MTL will achieve even better performance, suggesting that tasks are beneficial in a complementary manner and there is an advantage of MTL beyond two tasks.

*The opposite is almost true.* When pairwise MTL does not improve upon STL, most of the time All MTL will not improve upon STL, either — with one exception: COM. Specifically, the pairwise MTL performances of UPOS, XPOS, NER and FRAME (**TE⊕DEC**) are mostly negative and so are their All

| | UPOS | XPOS | CHUNK | NER | MWE | SEM | SEMTR | SUPSENSE | COM | FRAME | HYP | #↑ | #↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +UPOS | 95.4 | 94.94 | 94.0 ↑ | 87.43 ↓ | 57.61 ↑ | 73.11 ↑ | 74.85 ↑ | 67.76 ↑ | 72.09 | 62.27 | 48.27 | 5 | 1 |
| +XPOS | 95.42 | 95.04 | 93.98 ↑ | 87.71 ↓ | 58.26 ↑ | 73.04 | 74.66 ↑ | 67.77 ↑ | 72.41 | 61.62 | 48.06 ↑ | 5 | 1 |
| +CHUNK | 95.4 | 95.1 | 93.49 | 88.07 | 58.06 ↑ | 73.13 ↑ | 74.77 ↑ | 67.36 | 72.88 | 62.98 | 47.13 | 3 | 0 |
| +NER | 95.29 | 95.05 | 93.54 | 88.24 | 53.4 | 72.91 | 74.04 | 67.57 ↑ | 70.78 ↓ | 63.02 | 48.64 | 1 | 1 |
| +MWE | 95.05 ↓ | 94.66 ↓ | 93.33 | 88.02 | 53.07 | 72.83 | 74.66 ↑ | 66.26 | 71.36 | 60.61 | 46.71 | 1 | 2 |
| +SEM | 95.27 | 94.93 | 93.52 | 87.49 ↓ | 58.62 ↑ | 72.77 | 74.41 ↑ | 68.1 ↑ | 72.25 | 62.17 | 47.12 | 3 | 1 |
| +SEMTR | 95.23 | 94.97 | 93.45 | 87.29 ↓ | 58.31 ↑ | 72.17 ↓ | 74.02 | 67.64 | 72.15 | 62.79 | 46.1 | 1 | 2 |
| +SUPSENSE | 95.27 | 95.0 | 93.13 ↓ | 87.92 | 58.05 ↑ | 73.09 ↑ | 74.94 ↑ | 66.81 | 72.12 | 61.96 | 47.24 | 3 | 1 |
| +COM | 93.6 ↓ | 93.12 ↓ | 91.86 ↓ | 86.75 ↓ | 51.71 | 70.18 ↓ | 71.35 ↓ | 65.55 ↓ | 72.71 | 57.65 | 47.81 | 0 | 7 |
| +FRAME | 95.0 ↓ | 94.55 ↓ | 93.29 | 87.99 | 53.3 | 72.49 | 74.63 ↑ | 66.75 | 72.1 | 62.04 | 46.66 | 1 | 2 |
| +HYP | 94.43 ↓ | 94.26 ↓ | 93.13 ↓ | 87.82 | 52.59 | 71.95 | 74.14 | 66.16 | 72.79 | 61.14 | 46.73 | 0 | 3 |
| #↑ | 0 | 0 | 2 | 0 | 6 | 3 | 7 | 4 | 0 | 0 | 1 | | |
| #↓ | 4 | 4 | 3 | 5 | 0 | 2 | 1 | 1 | 1 | 0 | 0 | | |
| Average | 95.0 | 94.66 | 93.32 | 87.65 | 55.99 | 72.49 | 74.24 | 67.09 | 72.09 | 61.62 | 47.37 | | |
| All | 94.94 ↓ | 94.3 ↓ | 93.7 ↑ | 86.01 ↓ | 59.57 ↑ | 71.58 ↓ | 74.35 | 68.02 ↑ | 74.61 ↑ | 61.83 | 49.5 ↑ | 5 | 4 |
| Oracle | 95.4 | 95.04 | 93.93 ↑ | 88.24 | 61.92 ↑ | 73.14 ↑ | 75.09 ↑ | 69.04 ↑ | 72.71 | 62.04 | 48.06 ↑ | 6 | 0 |

Table 5: F1 scores for **TE⊕ENC**. We compare STL setting (blue), with pairwise MTL (+⟨task⟩), All, and Oracle. We test on each task in the columns. Beneficial settings are in green. Harmful setting are in red. The last two columns indicate how many tasks are helped or harmed by the task at that row.

| | UPOS | XPOS | CHUNK | NER | MWE | SEM | SEMTR | SUPSENSE | COM | FRAME | HYP | #↑ | #↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| All | 95.04 | 94.31 | 93.44 | 86.38 | 61.43 | 71.53 | 74.26 | 68.1 | 74.54 | 59.71 | 51.41 | | |
| All - UPOS | | 94.03 | 93.59 | 86.03 | 61.28 | 70.87 | 73.54 | 68.27 | 74.42 | 58.47 | 51.13 | 0 | 0 |
| All - XPOS | 94.57 ↓ | | 93.57 | 86.04 | 61.91 | 71.12 | 74.03 | 67.99 | 74.36 | 60.16 | 51.65 | 0 | 1 |
| All - CHUNK | 94.84 ↓ | 94.46 | | 86.05 | 61.01 | 71.07 | 73.97 | 68.26 | 74.2 | 60.01 | 50.27 | 0 | 1 |
| All - NER | 94.81 ↓ | 94.3 | 93.59 | | 62.69 | 70.82 | 73.51 ↓ | 68.16 | 74.08 | 59.17 | 50.86 | 0 | 2 |
| All - MWE | 94.93 ↓ | 94.45 | 93.71 | 86.21 | | 71.01 | 73.61 ↓ | 68.18 | 74.7 | 59.23 | 50.83 | 0 | 2 |
| All - SEM | 94.82 | 94.34 | 93.63 | 85.81 | 61.17 | | 71.97 ↓ | 67.36 | 74.31 | 58.73 | 50.93 | 0 | 1 |
| All - SEMTR | 94.83 | 94.35 | 93.58 | 86.11 | 63.04 | 69.72 ↓ | | 68.17 | 74.2 | 59.49 | 51.27 | 0 | 1 |
| All - SUPSENSE | 94.97 | 94.54 | 93.67 | 86.43 | 60.51 | 71.22 | 73.86 ↓ | | 74.24 | 59.23 | 50.86 | 0 | 1 |
| All - COM | 95.19 ↑ | 94.69 ↑ | 93.67 | 86.6 | 61.95 | 72.38 ↑ | 74.75 ↑ | 68.67 | | 62.37 ↑ | 50.28 | 5 | 0 |
| All - FRAME | 95.15 | 94.57 | 93.7 | 85.9 | 62.62 | 71.48 | 74.24 | 68.47 | 75.03 | | 50.89 | 0 | 0 |
| All - HYP | 94.93 | 94.53 | 93.78 ↑ | 86.31 | 62.04 | 71.22 | 74.02 | 68.46 | 74.62 | 59.69 | | 1 | 0 |
| #↑ | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | | |
| #↓ | 4 | 0 | 0 | 0 | 0 | 1 | 4 | 0 | 0 | 0 | 0 | | |

Table 6: F1 scores for **MULTI-DEC**. We compare All with All-but-one settings (All - ⟨TASK⟩). We test on each task in the columns. Beneficial settings are in green. Harmful setting are in red.

MTL performances. Furthermore, tasks can also be harmful in a complementary manner. For instance, in the case of NER, All MTL achieves the lowest or the second lowest score when compared to any row of the pairwise MTL settings. In addition, SEM's pairwise MTL performances are mixed, making the average score about the same or slightly worse than STL. However, the performance of All MTL when tested on SEM almost achieves the lowest. In other words, SEM is harmed more than it is benefited but pairwise MTL performances cannot tell. This suggests that harmful tasks are complementary while beneficial tasks are not.

Our results when *tested on* COM *are the most surprising*. While none of pairwise MTL settings help (with some hurting), the performance of All MTL goes in the opposite direction, outperforming that of STL. Further characterization of task interaction is needed to reveal why this happens. One hypothesis is that instances in COM that are benefited by one task may be harmed by another. The joint training of all tasks thus works because tasks *regularize* each other.

We believe that our results open the doors to other interesting research questions. While the pairwise MTL performance is somewhat predictive of the performance direction of All MTL (except COM), the magnitude of that direction is difficult to predict. It is clear that additional factors beyond pairwise performance contribute to the success or failure of the All MTL setting. It would be useful to automatically identify these factors or design a metric to capture that. There have been initial attempts along this research direction in (Alonso and Plank, 2017; Bingel and Søgaard, 2017; Bjerva, 2017), in which manually-defined task characteristics are found to be predictive of *pairwise* MTL's failure or success.

**Oracle MTL** Recall that a task has an "Oracle" set when the task is benefited from some other tasks according to its pairwise results (cf. Sect. 3.4). In general, our simple heuristic works well. Out of 20 cases where Oracle MTL performances exist, 16 are better than the performance of All MTL. In SEM, UPOS and XPOS (**TE⊕DEC**, Oracle MTL is able to reverse the negative results obtained by All MTL to the positive ones, leading to improved scores over STL in all cases. This suggests that pairwise MTL performances are valuable knowledge if we want to go beyond two tasks. But, as mentioned previously,

pairwise performance information fails in the case of COM; All MTL leads to improvement but we do not have an Oracle set in this case.

Out of 4 cases where Oracle MTL does not improve upon All MTL, 3 is when we test on HYP and one is when we test on MWE. These two tasks are not harmed by any tasks. This result seems to suggest that sometimes "neutral" tasks can help in MTL (but not always, for example, in **MULTI-DEC** and **TE⊕ENC** of MWE). This also raises the question of whether there is a more effective way to construct an oracle set.

### 4.4 Analysis

**Task Contribution in All MTL** How much does one particular task contribute to the performance of All MTL? To investigate this, we remove one task at a time and train the rest jointly. Results are shown in Table 6 for the method **MULTI-DEC**– results for other two methods are in the arXiv version as they are similar to **MULTI-DEC** qualitatively. We find that UPOS, SEM and SEMTR are in general sensitive to a task being removed from All MTL. Moreover, at least one task significantly contributes to the success of All MTL at some point; if we remove it, the performance will drop. On the other hand, COM generally negatively affects the performance of All MTL as removing it often leads to performance improvement.

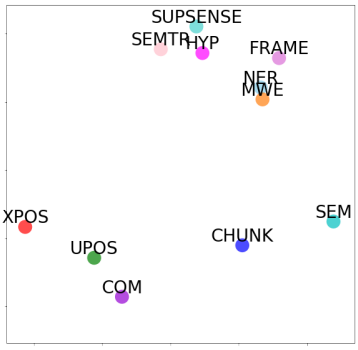



Figure 4: t-SNE visualization of the embeddings of the 11 tasks that are learned from **TE⊕DEC**

**Task Embeddings** Fig. 4 shows t-SNE visualization (Van der Maaten and Hinton, 2008) of task embeddings learned from **TE⊕DEC**[6] in the All MTL setting. The learned task embeddings reflect our knowledge about similarities between tasks, where there are clusters of syntactic and semantic tasks. We also learn that sentence compression (COM) is more syntactic, whereas multi-word expression identification (MWE) and hyper-text detection (HYP) are more semantic. Interestingly, CHUNK seems to be in between, which may explain why it never harms any tasks in any settings (cf. Sect. 4.2).

In general, it is not obvious how to translate task similarities derived from task embeddings into something indicative of MTL performance. While our task embeddings could be considered as "task characteristics" vectors, they are not guaranteed to be interpretable. We thus leave a thorough investigation of information captured by task embeddings to future work.

Nevertheless, we observe that task embeddings disentangle "sentences/tags" and "actual task" to some degree. For instance, if we consider the locations of each pair of tasks that use the same set of sentences for training in Fig. 4, we see that SEM and SEMTR (or MWE and SUPSENSE) are not neighbors, while XPOS and UPOS are. On the other hand, MWE and NER are neighbors, even though their label set size and entropy are not the closest. These observations suggest that hand-designed task features used in (Bingel and Søgaard, 2017) may not be the most informative characterization for predicting MTL performance.

## 5 Related Work

For a comprehensive overview of MTL in NLP, see Chapter 20 of (Goldberg, 2017) and (Ruder, 2017). Here we highlight those which are mostly relevant.

[6]We observed that task embeddings learned from **TE⊕ENC** are not consistent across multiple runs.

MTL for NLP has been popular since a unified architecture was proposed by (Collobert and Weston, 2008; Collobert et al., 2011). As for sequence to sequence learning (Sutskever et al., 2014), general multi-task learning frameworks are explored by (Luong et al., 2016).

Our work is different from existing work in several aspects. First, the majority of the work focuses on two tasks, often with one being the main task and the other being the auxiliary one (Søgaard and Goldberg, 2016; Bjerva et al., 2016; Plank et al., 2016; Alonso and Plank, 2017; Bingel and Søgaard, 2017). For example, POS is the auxiliary task in (Søgaard and Goldberg, 2016) while CHUNK, CCG supertagging (CCG) (Clark, 2002), NER, SEM, or MWE+SUPSENSE is the main one. They find that POS benefits CHUNK and CCG. Another line of work considers language modeling as the auxiliary objective (Godwin et al., 2016; Rei, 2017; Liu et al., 2018). Besides sequence tagging, some work considers two high-level tasks or one high-level task with another lower-level one. Examples are dependency parsing (DEP) with POS (Zhang and Weiss, 2016), with MWE (Constant and Nivre, 2016), or with semantic role labeling (SRL) (Shi et al., 2016); machine translation (TRANSLATE) with POS or DEP (Niehues and Cho, 2017; Eriguchi et al., 2017); sentence extraction and COM (Martins and Smith, 2009; Berg-Kirkpatrick et al., 2011; Almeida and Martins, 2013).

Exceptions to this include the work of (Collobert et al., 2011), which considers four tasks: POS, CHUNK, NER, and SRL; (Raganato et al., 2017), which considers three: word sense disambiguation with POS and coarse-grained semantic tagging based on WordNet lexicographer files; (Hashimoto et al., 2017), which considers five: POS, CHUNK, DEP, semantic relatedness, and textual entailment; (Niehues and Cho, 2017; Kiperwasser and Ballesteros, 2018), which both consider three: TRANSLATE with POS and NER, and TRANSLATE with POS and DEP, respectively. We consider as many as 11 tasks jointly.

Second, we choose to focus on model architectures that are generic enough to be shared by many tasks. Our structure is similar to (Collobert et al., 2011), but we also explore frameworks related to task embeddings and propose two variants. In contrast, recent work considers stacked architectures (mostly for sequence tagging) in which tasks can supervise at different layers of a network (Søgaard and Goldberg, 2016; Klerke et al., 2016; Plank et al., 2016; Alonso and Plank, 2017; Bingel and Søgaard, 2017; Hashimoto et al., 2017). More complicated structures require more sophisticated MTL methods when the number of tasks grows and thus prevent us from concentrating on analyzing relationships among tasks. For this reason, we leave MTL for complicated models for future work.

The purpose of our study is relevant to but different from transfer learning, where the setting designates one or more target tasks and focuses on whether the target tasks can be learned more effectively from the source tasks; see e.g., (Mou et al., 2016; Yang et al., 2017).

## 6 Discussion and Future Work

We conduct an empirical study on MTL for sequence tagging, which so far has been mostly studied with two or a few tasks. We also propose two alternative frameworks that augment taggers with task embeddings. Our results provide insights regarding task relatedness and show benefits of the MTL approaches. Nevertheless, we believe that our work simply scratches the surface of MTL. The characterization of task relationships seems to go beyond the performances of pairwise MTL training or similarities of their task embeddings. We are also interested in exploring further other techniques to MTL, especially when tasks become more complicated. For example, it is not clear how to best represent task specification as well as how to incorporate them into NLP systems. Finally, the definition of tasks can be relaxed to include domains or languages. Combining all these will move us toward the goal of having a *single* robust, generalizable NLP agent that is equipped with a diverse set of skills.

**Acknowledgments** This work is partially supported by USC Graduate Fellowship, NSF IIS-1065243, 1451412, 1513966/1632803/1833137, 1208500, CCF-1139148, a Google Research Award, an Alfred P. Sloan Research Fellowship, gifts from Facebook and Netflix, and ARO# W911NF-12-1-0241 and W911NF-15-1-0484.

## References


[Almeida and Martins2013] Miguel B. Almeida and André F. T. Martins. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *ACL*. 10

[Alonso and Plank2017] Héctor Martínez Alonso and Barbara Plank. 2017. Multitask learning for semantic sequence prediction under varying data conditions. In *EACL*. 1, 3, 8, 10

[Baker et al.1998] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *COLING-ACL*. 4

[Berg-Kirkpatrick et al.2011] Taylor Berg-Kirkpatrick, Daniel Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *ACL*. 10

[Bies et al.2012] Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English web treebank. Technical Report LDC2012T13, Linguistic Data Consortium, Philadelphia, PA. 3

[Bingel and Søgaard2017] Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *EACL*. 1, 4, 8, 9, 10

[Bjerva et al.2016] Johannes Bjerva, Barbara Plank, and Johan Bos. 2016. Semantic tagging with deep residual networks. In *COLING*. 1, 10

[Bjerva2017] Johannes Bjerva. 2017. Will my auxiliary tagging task help? Estimating Auxiliary Tasks Effectivity in Multi-Task Learning. In *NoDaLiDa*. 8

[Caruana1997] Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28:41–75. 1

[Cho et al.2014] Kyunghyun Cho, Bart van Merrienboer, aglar Gülehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*. 4

[Ciaramita and Altun2006] Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *EMNLP*. 3

[Clark2002] Stephen Clark. 2002. Supertagging for combinatory categorial grammar. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks*. 10

[Clarke and Lapata2006] James Clarke and Mirella Lapata. 2006. Constraint-based sentence compression: An integer programming approach. In *ACL*. 4

[Collobert and Weston2008] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*. 1, 2, 10

[Collobert et al.2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537. 1, 2, 10

[Constant and Nivre2016] Matthieu Constant and Joakim Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. In *ACL*. 10

[Das et al.2014] Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*, 40:9–56. 4

[Eriguchi et al.2017] Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *ACL*. 10

[Francis and Kučera1982] Winthrop Nelson Francis and Henry Kučera. 1982. Frequency analysis of english usage: Lexicon and grammar. *Journal of English Linguistics*, 18(1):64–70. 3

[Gardner et al.2018] Matt A. Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*. 4

[Glorot and Bengio2010] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*. 5

[Godwin et al.2016] Jonathan Godwin, Pontus Stenetorp, and Sebastian Riedel. 2016. Deep semi-supervised learning with linguistically motivated sequence labeling task hierarchies. *arXiv preprint arXiv:1612.09113*. 10

[Goldberg2017] Yoav Goldberg. 2017. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309. 9

[Graff1997] David Graff. 1997. The 1996 broadcast news speech and language-model corpus. In *Proceedings of the 1997 DARPA Speech Recognition Workshop*. 4

[Hashimoto et al.2017] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A Joint Many-Task Model: Growing a neural network for multiple NLP tasks. In *EMNLP*. 10

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–80. 4

[Huang et al.2015] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*. 4

[Irsoy and Cardie2014] Ozan Irsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *EMNLP*. 4

[Johnson et al.2017] Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *TACL*, 5:339–351. 3

[Kingma and Ba2015] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*. 5

[Kiperwasser and Ballesteros2018] Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *TACL*, 6:225–240. 10

[Klerke et al.2016] Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016. Improving sentence compression by learning to predict gaze. In *HLT-NAACL*. 10

[Lafferty et al.2001] John D. Lafferty, Andrew D. McCallum, and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*. 4

[Lample et al.2016] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *HLT-NAACL*. 4

[Landes et al.1998] Shari Landes, Claudia Leacock, and Randee I. Tengi. 1998. Building semantic concordances. *WordNet: An electronic lexical database*, 199(216):199–216. 3

[Lewis et al.2004] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397. 3

[Liu et al.2018] Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *AAAI*. 10

[Luong et al.2016] Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *ICLR*. 1, 2, 10

[Ma and Hovy2016] Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *ACL*. 4, 5

[Marcus et al.1993] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330. 3

[Martins and Smith2009] André F. T. Martins and Noah A. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the NAACL-HLT Workshop on Integer Linear Programming for NLP*. 10

[Miller et al.1993] George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology*. 3

[Mou et al.2016] 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*. 10

[Niehues and Cho2017] Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. In *WMT*. 10

[Nivre et al.2016] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *LREC*. 3

[Pascanu et al.2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *ICML*. 5

[Paszke et al.2017] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *Proceedings of the NIPS Workshop on the future of gradient-based machine learning software and techniques*. 4

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*. 5

[Plank et al.2016] Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *ACL*. 1, 10

[Raganato et al.2017] Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017. Neural sequence learning models for word sense disambiguation. In *EMNLP*. 10

[Rei2017] Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *ACL*. 10

[Reimers and Gurevych2017] Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *EMNLP*. 2

[Ruder2017] Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*. 9

[Schneider and Smith2015] Nathan Schneider and Noah A. Smith. 2015. A corpus and model integrating multiword expressions and supersenses. In *HLT-NAACL*. 3

[Shi et al.2016] Peng Shi, Zhiyang Teng, and Yue Zhang. 2016. Exploiting mutual benefits between syntax and semantic roles using neural network. In *EMNLP*. 10

[Søgaard and Goldberg2016] Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *ACL*. 1, 10

[Spitkovsky et al.2010] Valentin I. Spitkovsky, Daniel Jurafsky, and Hiyan Alshawi. 2010. Profiting from mark-up: Hyper-text annotations for guided parsing. In *ACL*. 4

[Sutskever et al.2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*. 2, 10

[Tjong Kim Sang and Buchholz2000] Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *CoNLL*. 3

[Tjong Kim Sang and De Meulder2003] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *CoNLL*. 3

[Van der Maaten and Hinton2008] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(2579-2605):85. 9

[Vossen et al.1998] Piek Vossen, Laura Bloksma, Horacio Rodriguez, Salvador Climent, Nicoletta Calzolari, Adriana Roventini, Francesca Bertagna, Antonietta Alonge, and Wim Peters. 1998. The EuroWordNet Base Concepts and Top Ontology. Technical Report LE2-4003, University of Amsterdam, The Netherlands. 3

[Yang et al.2017] Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *ICLR*. 10

[Zhang and Weiss2016] Yuan Zhang and David Weiss. 2016. Stack-propagation: Improved representation learning for syntax. In *ACL*. 10