A Joint Named-Entity Recognizer for Heterogeneous Tag-sets Using a Tag Hierarchy

Genady Beryozkin, Yoel Drori, Oren Gilon, Tzvika Hartman and Idan Szpektor
Google Research
Tel Aviv, Israel
{genady,dyoel,ogilon,tzvika,szpektor} @google.com

Abstract

We study a variant of domain adaptation for named-entity recognition where multiple, heterogeneously tagged training sets are available. Furthermore, the test tag-set is not identical to any individual training tag-set. Yet, the relations between all tags are provided in a tag hierarchy, covering the test tags as a combination of training tags. This setting occurs when various datasets are created using different annotation schemes. This is also the case of extending a tag-set with a new tag by annotating only the new tag in a new dataset. We propose to use the given tag hierarchy to jointly learn a neural network that shares its tagging layer among all tag-sets. We compare this model to combining independent models and to a model based on the multitasking approach. Our experiments show the benefit of the tag-hierarchy model, especially when facing non-trivial consolidation of tag-sets.

1 Introduction

Named Entity Recognition (NER) has seen significant progress in the last couple of years with the application of Neural Networks to the task. Such models achieve state-of-the-art performance with little or no manual feature engineering (Collobert et al., 2011; Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016; Dernoncourt et al., 2017). Following this success, more complex NER setups are approached with neural models, among them domain adaptation (Qu et al., 2016; He and Sun, 2017; Dong et al., 2017).

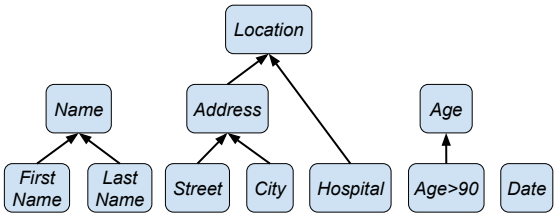
In this work we study one type of domain adaptation for NER, denoted here *heterogeneous tag-sets*. In this variant, samples from the test set are not available at training time. Furthermore, the test tag-set differs from each training tag-set. However every test tag can be represented either as a single training tag or as a combination of several training tags. This information is given in the form of a hypernym hierarchy over all tags, training and test (see Fig. 1).

This setting arises when different schemes are used for annotating multiple datasets for the same task. This often occurs in the medical domain, where healthcare providers use customized tag-sets to create their own private test sets (Shickel et al., 2017; Lee et al., 2018). Another scenario is selective annotation, as in the case of extending an existing tag-set, *e.g.* {'*Name*', '*Location*'}, with another tag, *e.g.* '*Date*'. To save annotation effort, new training data is labeled only with the new tag. This case of disjoint tag-sets is also discussed in the work of Greenberg et al. (2018). A similar case is extending a training-set with new examples in which only rare tags are annotated. In domains where training data is scarce, out-of-domain datasets annotated with infrequent tags may be very valuable.

A naive approach concatenates all training-sets, ignoring the differences between the tagging schemes in each example. A different approach would be to learn to tag with multiple training tag-sets. Then, in a post-processing step, the predictions from the different tag-sets need to be consolidated into a single test tag sequence, resolving tagging differences along the way. We study two such models. The first model learns an independent NER model for each training tag-set. The second model applies the multitasking (MTL) (Collobert et al., 2011; Ruder, 2017) paradigm, in which a shared latent representation of the input text is fed into separate tagging layers.

The above models require heuristic post-processing to consolidate the different predicted tag sequences. To overcome this limitation, we propose a model that incorporates the given tag hierarchy within the neural NER model. Specifically, this model learns to predict a tag sequence only over the fine-grained tags in the hierarchy.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 140–150
Florence, Italy, July 28 - August 2, 2019.



Tag-set 1 (T_1): *Name, Street, City, Hospital, Age>90*
Tag-set 2 (T_2): *First Name, Last Name, Address, Age*
Tag-set 3 (T_3): *Name, Location, Date*

Figure 1: A tag hierarchy for three tag-sets.

At training time, gradients on each dataset-specific labeled examples are propagated as gradients on plausible fine-grained tags. At inference time the model predicts a single sequence of fine-grained tags, which are then mapped to the test tag-set by traversing the tag hierarchy. Importantly, all tagging decisions are performed in the model without the need for a post-processing consolidation step.

We conducted two experiments. The first evaluated the extension of a tag-set with a new tag via selective annotation of a new dataset with only the extending tag, using datasets from the medical and news domains. In the second experiment we integrated two full tag-sets from the medical domain with their training data while evaluating on a third test tag-set. The results show that the model which incorporates the tag-hierarchy is more robust compared to a combination of independent models or MTL, and typically outperforms them. This is especially evident when many tagging collisions need to be settled at post-processing. In these cases, the performance gap in favor of the tag-hierarchy model is large.

2 Background and Definitions

2.1 Task Definition

The goal in the *heterogeneous tag-sets* domain adaptation task is to learn an NER model M that given an input token sequence $x = \{x_i\}_1^n$ infers a tag sequence $y = \{y_i\}_1^n = M(x)$ over a test tag-set T^s, $\forall_i\, y_i \in T^s$. To learn the model, K training datasets $\{DS_k^r\}_{k=1}^K$ are provided, each labeled with its own tag-set T_k^r. Superscripts 's' and 'r' stand for 'test' and 'training', respectively. In this task, no training tag-set is identical to the test tag-set T^s by itself. However, all tags in T^s can be covered by combining the training tag-sets $\{T_k^r\}_{k=1}^K$. This information is provided in the form of a directed acyclic graph (DAG) representing hypernymy relations between all training and test tags. Fig. 1 illustrates such a hierarchy.

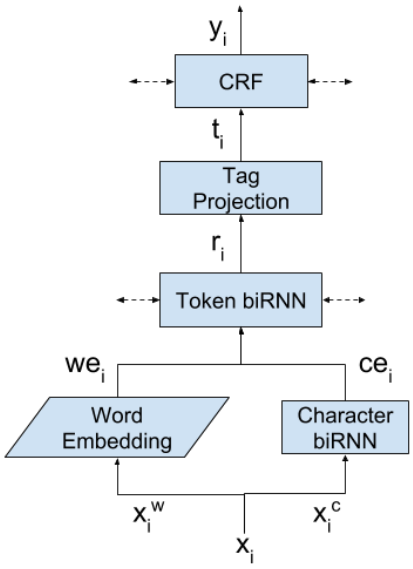


Figure 2: Neural architecture for NER.

As mentioned above, an example scenario is selective annotation, in which an original tag-set is extended with a new tag t, each with its own training data, and the test tag-set is their union. But, some setups require combinations other than a simple union, *e.g.* covering the test tag '*Address*' with the finer training tags '*Street*' and '*City*', each from a different tag-set.

This task is different from inductive domain adaptation (Pan and Yang, 2010; Ruder, 2017), in which the tag-sets are different but the tasks differ as well (*e.g.* NER and parsing), with no need to map the outcomes to a single tag-set at test time.

2.2 Neural network for NER

As the underlying architecture shared by all models in this paper, we follow the neural network proposed by Lample et al. (2016), which achieved state-of-the-art results on NER. In this model, depicted in Fig. 2, each input token x_i is represented as a combination of: (a) a one-hot vector x_i^w, mapping the input to a fixed word vocabulary, and (b) a sequence of one-hot vectors $\{x_{i,j}^c\}_{j=1}^{n_i}$, representing the input word's character sequence.

Each input token x_i is first embedded in latent space by applying both a word-embedding matrix, $we_i = E\, x_i^w$, and a character-based embedding layer $ce_i = \text{CharBiRNN}(\{x_{i,j}^c\})$ (Ling et al., 2015). This output of this step is $e_i = ce_i \oplus we_i$, where $\oplus$ stands for vector concatenation. Then, the embedding vector sequence $\{e_i\}_1^n$

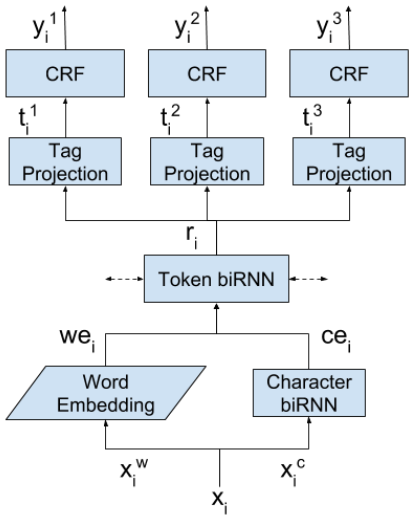


Figure 3: NER multitasking architecture for 3 tag-sets.

is re-encoded in context using a bidirectional RNN layer $\{r_i\}_1^n = \text{BiRNN}(\{e_i\}_1^n)$ (Schuster and Paliwal, 1997). The sequence $\{r_i\}_1^n$ constitutes the latent representation of the input text.

Finally, each re-encoded vector r_i is projected to tag space for the target tag-set T, $t_i = P r_i$, where $|t_i| = |T|$. The sequence $\{t_i\}_1^n$ is then taken as input to a CRF layer (Lafferty et al., 2001), which maintains a global tag transition matrix. At inference time, the model output is $y = M(x)$, the most probable CRF tag sequence for input x.

3 Models for Multiple Tagging Layers

One way to learn a model for the heterogeneous tag-sets setting is to train a base NER (Sec. 2.2) on the concatenation of all training-sets, predicting tags from the union of all training tag-sets. In our experiments, this model under performed, due to the fact that it treats each training example as fully tagged despite being tagged only with the tags belonging to the training-set from which the example is taken (see Sec. 6).

We next present two models that instead learn to tag each training tag-set separately. In the first model the outputs from independent base models, each trained on a different tag-set, are merged. The second model utilizes the the multitasking approach to train separate tagging layers that share a single text representation layer.

3.1 Combining independent models

In this model, we train a separate NER model for each training set, resulting in K models $\{M_k\}_{k=1}^K$. At test time, each model predicts a sequence $y_k = M_k(x)$ over the corresponding tag-set T_k^r. The sequences $\{y_k\}_{k=1}^K$ are consolidated into a single sequence y^s over the test tag-set T^s.

We perform this consolidation in a post-processing step. First, each predicted tag $y_{k,i}$ is mapped to the test tag-set as $y_{k,i}^s$. We employ the provided tag hierarchy for this mapping by traversing it starting from $y_{k,i}$ until a test tag is reached. Then, for every token x_i, we consider the test tags predicted at position i by the different models $M(x_i) = \{y_{k,i}^s | y_{k,i}^s \neq \text{`}Other\text{'}\}$. Cases where $M(x_i)$ contains more than one tag are called *collisions*. Models must consolidate collisions, selecting a single predicted tag for x_i.

We introduce three different consolidation methods. The first is to randomly select a tag from $M(x_i)$. The second chooses the tag that originates from the tag sequence y_k with the highest CRF probability score. The third computes the marginal CRF tag probability for each tag and selects the one with the highest probability.

3.2 Multitasking for heterogeneous tag-sets

Lately, several works explored using multitasking (MTL) for inductive transfer learning within a neural architecture (Collobert and Weston, 2008; Chen et al., 2016; Peng and Dredze, 2017). Such algorithms jointly train a single model to solve different NLP tasks, such as NER, sentiment analysis and text classification. The various tasks share the same text representation layer in the model but maintain a separate tagging layer per task.

We adapt multitasking to heterogeneous tag-sets by considering each training dataset, which has a different tag-set T_k^r, as a separate NER task. Thus, a single model is trained, in which the latent text representation $\{r_i\}_1^n$ (see Sec. 2.2) is shared between NER tasks. As mentioned above, the tagging layers (projection and CRF) are kept separate for each tag-set. Fig. 3 illustrates this architecture.

We emphasize that the output of the MTL model still consists of $\{y_k\}_{k=1}^K$ different tag sequence predictions. They are consolidated into a final single sequence y^s using the same post-processing step described in Sec. 3.1.

4 Tag Hierarchy Model

The models introduced in Sec. 3.1 and 3.2 learn to predict a tag sequence for each training tag-set separately and they do not share parameters between tagging layers. In addition, they require

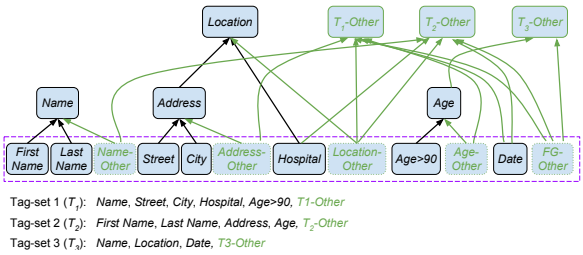


Figure 4: The tag hierarchy in Fig. 1 for three tag-sets after closure extension. Green nodes and edges were automatically added in this process. Fine-grained tags are surrounded by a dotted box.

a post-processing step, outside of the model, for merging the tag sequences inferred for the different tag-sets. A simple concatenation of all training data is also not enough to accommodate the differences between the tag-sets within the model (see Sec. 3). Moreover, none of these models utilizes the relations between tags, which are provided as input in the form of a tag hierarchy.

In this section, we propose a model that addresses these limitations. This model utilizes the given tag hierarchy at training time to learn a single, shared tagging layer that predicts only fine-grained tags. The hierarchy is then used during inference to map fine-grained tags onto a target tag-set. Consequently, all tagging decisions are made in the model, without the need for a post-processing step.

4.1 Notations

In the input hierarchy DAG, each node represents some semantic role of words in sentences, (e.g. '*Name*'). A directed edge $c \rightarrow d$ implies that c is a hyponym of d, meaning c captures a subset of the semantics of d. Examples include '$LastName$' $\rightarrow$ '$Name$', and '$Street$' $\rightarrow$ '$Location$' in Fig. 1. We denote the set of all tags that capture some subset of semantics of d by $\mathrm{Sem}(d) = \{d\} \cup \{c | c \xrightarrow{\mathrm{R}} d\}$, where $\xrightarrow{\mathrm{R}}$ indicates that there is a directed path from c to d in the graph. For example, $\mathrm{Sem}(Name) = \{Name, LastName, FirstName\}$.

If a node d has no hyponyms ($\mathrm{Sem}(d) = \{d\}$), it represents some fine-grained tag semantics. We denote the set of all fine-grained tags by T^{FG}. We also denote all fine-grained tags that are hyponyms of d by $\mathrm{Fine}(d) = T^{FG} \cap \mathrm{Sem}(d)$, *e.g.* $\mathrm{Fine}(Name) = \{LastName, FirstName\}$. As mentioned above, our hierarchical model predicts tag sequences only from T^{FG} and then maps them onto a target tag-set.

4.2 Hierarchy extension with 'Other' tags

For each tag d we would like the semantics captured by the union of semantics of all tags in $\mathrm{Fine}(d)$ to be exactly the semantics of d, making sure we will not miss any aspect of d when predicting only over T^{FG}. Yet, this semantics-equality property does not hold in general. One such example in Fig. 4 is '*Age*>90'$\rightarrow$'*Age*', because there may be age mentions below 90 annotated in T_2's dataset.

To fix the semantics-equality above, we use the notion of the '*Other*' tag in NER, which has the semantics of "*all the rest*". Specifically, for every $d \notin T^{FG}$, a fine-grained tag 'd-*Other*' $\in T^{FG}$ and an edge 'd-*Other*'$\rightarrow$'d' are automatically added to the graph, hence 'd-*Other*'$\in \mathrm{Fine}(d)$. For instance, '*Age-Other*'$\rightarrow$'*Age*'. These new tags represent the aspects of d not captured by the other tags in $\mathrm{Fine}(d)$.

Next a tag 'T_i-*Other*' is automatically added to each tag-set T_i, explicitly representing the "*all the rest*" semantics of T_i. The labels for 'T_i-*Other*' are induced automatically from unlabeled tokens in the original DS_i^r dataset. To make sure that the semantics-equality property above also holds for 'T_i-*Other*', a fine-grained tag '*FG-Other*' is also added, which captures the "*all the rest*" semantics at the fine-grained level. Then, each 'T_i-*Other*' is connected to all fine-grained tags that do not capture some semantics of the tags in T_i, defining:

$$\mathrm{Fine}(T_i\text{-Other}) = T^{FG} \setminus \bigcup_{d \in T_i \setminus \{T_i\text{-Other}\}} \mathrm{Sem}(d)$$

This mapping is important at training time, where 'T_i-*Other*' labels are used as distant supervision over their related fine-grained tags (Sec. 4.3). Fig. 4 depicts our hierarchy example after this step. We emphasize that all extensions in this step are done automatically as part of the model's algorithm.

4.3 NER model with tag hierarchy

One outcome of the extension step is that the set of fine-grained tags T^{FG} covers all distinct fine-grained semantics across all tag-sets. In the following, we train a single NER model (Sec. 2.2) that predicts sequences of tags from the T^{FG} tag-set. As there is only one tagging layer, model parameters are shared across all training examples.

At inference time, this model predicts the most likely fine-grained tag sequence y^{fg} for the input

x. As the model outputs only a single sequence, post-processing consolidation is not needed. The tag hierarchy is used to map each predicated fine-grained tag y_i^{fg} to a tag in a test tag-set T^s by traversing the out-edges of y_i^{fg} until a tag in T^s is reached. This procedure is also used in the baseline models (see Sec. 3.1) for mapping their predictions onto the test tag-set. However, unlike the baselines, which end with multiple candidate predictions in the test tag-set and need to consolidate between them, here, only a single fine-grained tag sequence is mapped, so no further consolidation is needed.

At training time, each example x that belongs to some training dataset DS_i^r is labeled with a gold-standard tag sequence y where the tags are taken only from the corresponding tag-set T_i^r. This means that tags $\{y_i\}$ are not necessarily fine-grained tags, so there is no direct supervision for predicting fine-grained tag sequences. However, each gold label y_i provides distant supervision over its related fine-grained tags, $\mathrm{Fine}(y_i)$. It indicates that one of them is the correct fine-grained label without explicitly stating which one, so we consider all possibilities in a probabilistic manner.

Henceforth, we say that a fine-grained tag sequence y^{fg} *agrees with* y if $\forall_i\ y_i^{fg} \in \mathrm{Fine}(y_i)$, *i.e.* y^{fg} is a plausible interpretation for y at the fine-grained tag level. For example, following Fig. 4, sequences ['*Hospital*', '*City*'] and ['*Street*', '*City*'] agree with ['*Location*', '*Location*'], unlike ['*City*', '*Last Name*']. We denote all fine-grained tag sequences that agree with y by $\mathrm{AgreeWith}(y)$.

Using this definition, the tag-hierarchy model is trained with the loss function:

$$loss(y) = -log(\frac{Z_y}{Z}) \quad (1)$$

$$Z_y = \sum_{y^{fg} \in \mathrm{AgreeWith}(y)} \phi(y^{fg}) \quad (2)$$

$$Z = \sum_{y^{fg}} \phi(y^{fg}) \quad (3)$$

where $\phi(y)$ stands for the model's score for sequence y, viewed as unnormalized probability. Z is the standard CRF partition function over all possible fine-grained tag sequences. Z_y, on the other hand, accumulates scores only of fine-grained tag sequences that agree with y. Thus, this loss function aims at increasing the summed probability of all fine-grained sequences agreeing with y. Both Z_y and Z can be computed efficiently using the Forward-Backward algorithm (Lafferty et al., 2001).

We note that we also considered finding the most likely tag sequence over a test tag-set at inference time by summing the probabilities of all fine-grained tag sequences that agree with each candidate sequence y: $\max_y \sum_{y^{fg} \in \mathrm{AgreeWith}(y)} \phi(y^{fg})$. However, this problem is NP-hard (Lyngsø and Pedersen, 2002). We plan to explore other alternatives in future work.

Dataset	Tag-set Size	# Tokens	Tagged Tokens (%)
I2B2'06 (train)	7	387,126	4.6
I2B2'06 (test)		163,488	4.2
I2B2'14 (train)	17	336,422	4.4
I2B2'14 (dev)		152,895	5.0
I2B2'14 (test)		316,212	4.6
Physio (test)	6	335,383	0.7
Conll (train)	4	203,621	16.7
Conll (dev)		51,362	16.7
Conll (test)		46,435	18.1
Onto (train)	18	1,304,491	13.1
Onto (test)		162,971	14.2

Table 1: Dataset statistics. *Tokens tagged* refer to percentage of tokens tagged not as '*Other*'.

5 Experimental Settings

To test the tag-hierarchy model under heterogeneous tag-set scenarios, we conducted experiments using datasets from two domains. We next describe these datasets as well as implementation details for the tested models. Sec. 6 then details the experiments and their results.

5.1 Datasets

Five datasets from two domains, *medical* and *news*, were used in our experiments. Table 1 summarizes their main statistics.

For the medical domain we used the datasets I2B2-2006 (denoted ***I2B2'06***) (Uzuner et al., 2007), I2B2-2014 (denoted ***I2B2'14***) (Stubbs and Uzuner, 2015) and the PhysioNet golden set (denoted ***Physio***) (Goldberger et al., 2000). These datasets are all annotated for the NER task of de-identification (a.k.a text anonymization) (Dernoncourt et al., 2017). Still, as seen in Table 1, each dataset is annotated with a different tag-set. Both ***I2B2'06*** and ***I2B2'14*** include train and test sets, while ***Physio*** contains only a test set.

For the news domain we used the English part of CONLL-2003 (denoted ***Conll***) (Tjong Kim Sang and De Meulder, 2003) and OntoNotes-v5 (denoted ***Onto***) (Weischedel et al., 2013), both with train and test sets. We note that ***I2B2'14***, ***Conll***

and ***Onto*** also contain a dev-set, which is used for hyper-param tuning (see below).

In all experiments, each example is a full document. Each document is split into tokens on white-spaces and punctuation. A tag-hierarchy covering the 57 tags from all five datasets was given as input to all models in all experiments. We constructed this hierarchy manually. The only non-trivial tag was '*Location*', which in ***I2B2'14*** is split into finer tags ('*City*', '*Street*' etc.) and includes also hospital mentions in ***Conll*** and ***Onto***. We resolved these relations similarly to the graph in Figure 1.

5.2 Compared Models

Four models were compared in our experiments:

$\boldsymbol{M_{Concat}}$ A single NER model on the concatenation of datasets and tag-sets (Sec. 3).

$\boldsymbol{M_{Indep}}$ Combining predictions of independent NER models, one per tag-set (Sec. 3.1).

$\boldsymbol{M_{MTL}}$ Multitasking over training tag-sets (Sec. 3.2).

$\boldsymbol{M_{Hier}}$ A tag hierarchy employed within a single base model (Sec. 4).

All models are based on the neural network described in Sec. 2.2. We tuned the hyper-params in the base model to achieve state-of-the-art results for a single NER model on ***Conll*** and ***I2B2'14*** when trained and tested on the same dataset (Strubell et al., 2017; Dernoncourt et al., 2017) (see Table 2). This is done to maintain a constant baseline, and is also due to the fact that ***I2B2'06*** does not have a standard dev-set.

We tuned hyper-params over the dev-sets of ***Conll*** and ***I2B2'14***. For character-based embedding we used a single bidirectional LSTM (Hochreiter and Schmidhuber, 1997) with hidden state size of 25. For word embeddings we used pre-trained GloVe embeddings[1] (Pennington et al., 2014), without further training. For token recoding we used a two-level stacked bidirectional LSTM (Graves et al., 2013) with both output and hidden state of size 100.

Once these hyper-params were set, no further tuning was made in our experiments, which means all models for heterogeneous tag-sets were tested under the above fixed hyper-param set. In each experiment, each model was trained until convergence on the respective training set.

[1] nlp.stanford.edu/data/glove.6B.zip

	I2B2'06	***I2B2'14***	***Conll***	***Onto***
Micro avg. F1	0.894	0.960	0.926	0.896

Table 2: F1 for training and testing a single base NER model on the same dataset.

	Tag Frequency in training / test (%)			
	I2B2'06	***I2B2'14***	***Conll***	***Onto***
Name	1.4 / 1.3	1.0 / 1.0	4.3 / 4.9	3.1 / 2.9
Date	1.7 / 1.5	2.4 / 2.5	0 / 0	2.7 / 3.1
Location	0.1 / 0.1	0.2 / 0.3	3.2 / 3.4	2.7 / 3.2
Hospital	0.6 / 0.7	0.3 / 0.3	0 / 0	0 / 0

Table 3: Occurrence statistics for tags used in the tag-set extension experiment, reported as % out of all tokens in the training and test sets of each dataset.

6 Experiments and Results

We performed two experiments. The first refers to selective annotation, in which an existing tag-set is extended with a new tag by annotating a new dataset only with the new tag. The second experiment tests the ability of each model to integrate two full tag-sets.

In all experiments we assess model performance via micro-averaged tag F1, in accordance with CoNLL evaluation (Tjong Kim Sang and De Meulder, 2003). Statistical significance was computed using the Wilcoxon two-sided signed ranks test at $p = 0.01$ (Wilcoxon, 1945). We next detail each experiment and its results.

In all our experiments, we found the performance of the different consolidation methods (Sec. 3.1) to be on par. One reason that using model scores does not beat random selection may be due to the overconfidence of the tagging models – their prediction probabilities are close to 0 or 1. We report figures for random selection as representative of all consolidation methods.

6.1 Tag-set extension experiment

In this experiment, we considered the 4 most frequent tags that occur in at least two of our datasets: '*Name*', '*Date*', '*Location*' and '*Hospital*' (Table 3 summarizes their statistics). For each frequent tag t and an ordered pair of datasets in which t occurs, we constructed new training sets by removing t from the first training set (termed *base dataset*) and remove all tags but t from the second training set (termed *extending dataset*). For example, for the *triplet* of { '*Name*', ***I2B2'14***, ***I2B2'06***}, we constructed a version of ***I2B2'14*** without '*Name*' annotations and a version of ***I2B2'06*** containing only annotations for '*Name*'. This process yielded 32 such triplets.

F1 AVERAGE Extending Tag	Base Dataset	Model Hier	Indep	MTL
Date	***I2B2'14***	**0.806**	0.795	0.787
	I2B2'06	0.756	0.761	**0.787**
	Onto	**0.835**	0.828	0.819
Date Total		**0.799**	0.795	0.798
Hospital	***I2B2'14***	0.931	**0.941**	0.918
	I2B2'06	**0.867**	0.866	0.853
Hospital Total		0.899	**0.904**	0.885
Location	***Conll***	**0.801**	0.784	0.793
	I2B2'14	**0.953**	0.913	0.905
	I2B2'06	**0.877**	0.848	0.820
	Onto	**0.785**	0.694	0.692
Location Total		**0.854**	0.810	0.802
Name	***Conll***	**0.847**	0.759	0.729
	I2B2'14	**0.918**	0.880	0.902
	I2B2'06	0.740	**0.743**	0.729
	Onto	**0.878**	0.862	0.862
Name Total		**0.846**	0.811	0.806
Grand Total		**0.854**	0.823	0.816

Table 4: F1 in the tag-set extension experiment, averaged over extending datasets for every base dataset.

For every triplet, we train all tested models on the two modified training sets and test them on the test-set of the base dataset (***I2B2'14*** in the example above). Each test-set was not altered and contains all tags of the base tag-set, including t.

$\boldsymbol{M_{Concat}}$ performed poorly in this experiment. For example, on the dataset extending ***I2B2'14*** with '*Name*' from ***I2B2'06***, $\boldsymbol{M_{Concat}}$ tagged only one '*Name*' out of over 4000 '*Name*' mentions in the test set. Given this, we do not provide further details of the results of $\boldsymbol{M_{Concat}}$ in this experiment.

For the three models tested, this experiment yields 96 results. The main results[2] of this experiment are shown in Table 4. Surprisingly, in more tests $\boldsymbol{M_{Indep}}$ outperformed $\boldsymbol{M_{MTL}}$ than vice versa, adding to prior observations that multitasking can hurt performance instead of improving it (Bingel and Søgaard, 2017; Alonso and Plank, 2017; Bjerva, 2017). But, applying a shared tagging layer on top of a shared text representation boosts the model's capability and stability. Indeed, overall, $\boldsymbol{M_{Hier}}$ outperforms the other models in most tests, and in the rest it is similar to the best performing model.

Analyzing the results, we noticed that the gap between model performance increases when more collisions are encountered for $\boldsymbol{M_{MTL}}$ and $\boldsymbol{M_{Indep}}$ at post-processing time (see Sec. 3.1). The amount of collisions may be viewed as a predictor for the baselines' difficulty to handle a specific heterogeneous tag-sets setting. Table 5 presents the tests in which more than 100 collisions were detected for either $\boldsymbol{M_{Indep}}$ or $\boldsymbol{M_{MTL}}$, constituting 66% of all test triplets. In these tests, $\boldsymbol{M_{Hier}}$ is a clear winner, outperforming the compared models in all but two comparisons, often by a significant margin.

[2]Detailed results for all 96 tests are given in the Appendix.

F1 Tag	Base	Extending	Model Hier	Indep	MTL
Date	***I2B2'14***	***I2B2'06***	0.899		*__0.903__
		Onto	*__0.713__	0.686	0.671
	I2B2'06	***Onto***	0.641	*__0.681__	
	Onto	***I2B2'06***	*__0.834__		0.807
Location	***Conll***	***I2B2'14***	*__0.818__	0.783	
		I2B2'06	*__0.748__		0.730
		Onto	*__0.836__	0.830	
	I2B2'14	***Conll***	*__0.954__	0.899	0.887
		Onto	*__0.951__	0.921	0.907
	I2B2'06	***Conll***	__0.876__	0.816	0.760
		Onto	*__0.869__	0.847	0.812
	Onto	***Conll***	*__0.747__	0.701	0.703
		I2B2'14	__0.793__	0.691	0.707
		I2B2'06	*__0.814__	0.691	
Name	***Conll***	***I2B2'14***	*__0.855__		0.690
		I2B2'06	*__0.827__	0.666	0.631
		Onto	__0.860__	0.841	
	I2B2'14	***Conll***	*__0.900__	0.863	
		I2B2'06	*__0.943__	0.893	
		Onto	*__0.911__	0.882	0.891
	I2B2'06	***Conll***	*__0.662__		0.653
	Onto	***Conll***	*__0.895__	0.888	
		I2B2'14	*__0.892__	0.872	
		I2B2'06	*__0.846__	0.827	

Table 5: F1 for tag-set extensions with more than 100 collisions. Blank entries indicate fewer than 100 collisions. (*) indicates all results that are statistically significantly better than others in that row.

F1 Tag	Base	Extending	Model Hier	Indep	MTL
Location	***I2B2'14***	***I2B2'06***	0.953	0.919	0.919
		Onto	0.954	0.899	0.887
Name	***Conll***	***I2B2'06***	0.846	0.827	0.809
		Onto	0.895	0.888	0.890

Table 6: Examples for performance differences when base datasets are extended with an in-domain dataset compared to an out-of-domain dataset.

Finally, we compared the models trained with selective annotation to an "upper-bound" of training and testing a single NER model on the same dataset with all tags annotated (Table 2). As expected, performance is usually lower with selective annotation. But, the drop intensifies when the base and extending datasets are from different domains – medical and news. In these cases, we observed that $\boldsymbol{M_{Hier}}$ is more robust. Its drop compared to combining datasets from the same domain is the least in almost all such combinations. Table 6 provides some illustrative examples.

6.2 Full tag-set integration experiment

A scenario distinct from selective annotation is the integration of full tag-sets. On one hand, more training data is available for similar tags. On the other hand, more tags need to be consolidated among the tag-sets.

F1	Test Set		
Model	***I2B2'06***	***I2B2'14***	***Physio***
I2B2'06	*0.894	0.730	0.637
I2B2'14	0.714	***0.960**	0.712
M_{Concat}	0.827	0.809	0.621
M_{Indep}	0.760	0.861	0.640
M_{MTL}	0.81	0.862	*0.739
M_{Hier}	***0.900**	*0.958	***0.760**

Collisions	Test Set		
	I2B2'06	***I2B2'14***	***Physio***
M_{Indep}	224	1272	114
M_{MTL}	158	584	44

Table 7: F1 for combining ***I2B2'06*** and ***I2B2'14***. The top two models were trained only on a single dataset. The lower table part holds the number of collisions at post-processing. (*) indicates results that are statistically significantly better than others in that column.

To test this scenario, we trained the tested model types on the training sets of ***I2B2'06*** and ***I2B2'14***, which have different tag-sets. The models were evaluated both on the test sets of these datasets and on ***Physio***, an unseen test-set that requires the combination of the two training tag-sets for full coverage of its tag-set. We also compared the models to single models trained on each of the training sets alone. Table 7 displays the results.

As expected, single models do well on the test-set companion of their training-set but they underperform on the other test-sets. This is expected because the tag-set on which they were trained does not cover well the tag-sets in the other test-sets.

When compared with the best-performing single model, using M_{Concat} shows reduced results on all 3 test sets. This can be attributed to reduced performance for types that are semantically different between datasets (e.g. '*Date*'), while performance on similar tags (e.g. '*Name*') does not drop.

Combining the two training sets using either M_{Indep} or M_{MTL} leads to substantial performance drop in 5 out of 6 test-sets compared to the best-performing single model. This is strongly correlated with the number of collisions encountered (see Table 7). Indeed, the only competitive result, M_{MTL} tested on ***Physio***, had less than 100 collisions. This demonstrates the non triviality in real-world tag-set integration, and the difficulty of resolving tagging decisions across tag-sets.

By contrast, M_{Hier} has no performance drop compared to the single models trained and tested on the same dataset. Moreover, it is the best performing model on the unseen ***Physio*** test-set, with 6% relative improvement in F1 over the best single model. This experiment points up the robustness of the tag hierarchy approach when applied to this heterogeneous tag-set scenario.

7 Related Work

Collobert et al. (2011) introduced the first competitive NN-based NER that required little or no feature engineering. Huang et al. (2015) combined LSTM with CRF, showing performance similar to non-NN models. Lample et al. (2016) extended this model with character-based embeddings in addition to word embedding, achieving state-of-the-art results. Similar architectures, such as combinations of convolutional networks as replacements of RNNs were shown to out-perform previous NER models (Ma and Hovy, 2016; Chiu and Nichols, 2016; Strubell et al., 2017).

Dernoncourt et al. (2017) and Liu et al. (2017) showed that the LSTM-CRF model achieves state-of-the-art results also for de-identification in the medical domain. Lee et al. (2018) demonstrated how performance drops significantly when the LSTM-CRF model is tested under transfer learning within the same domain in this task.

Collobert and Weston (2008) introduced MTL for NN, and other works followed, showing it helps in various NLP tasks (Chen et al., 2016; Peng and Dredze, 2017). Søgaard and Goldberg (2016) and Hashimoto et al. (2017) argue that cascading architectures can improve MTL performance. Several works have explored conditions for successful application of MTL (Bingel and Søgaard, 2017; Bjerva, 2017; Alonso and Plank, 2017).

Few works attempt to share information across datasets at the tagging level. Greenberg et al. (2018) proposed a single CRF model for tagging with heterogeneous tag-sets but without a hierarchy. They show the utility of this method for in-domain datasets with a balanced tag distribution. Our model can be viewed as an extension of theirs for tag hierarchies. Augenstein et al. (2018) use tag embeddings in MTL to further propagate information between tasks. Li et al. (2017) propose to use a tag-set made of cross-product of two different POS tag-sets and train a model for it. Given the explosion in tag-set size, they introduce automatic pruning of cross-product tags. Kim et al. (2015) and Qu et al. (2016) automatically learn correlations between tag-sets, given training data for both tag-sets. They rely on similar contexts for related source and target tags, such as '*professor*' and '*student*'.

Our tag-hierarchy model was inspired by recent work on hierarchical multi-label classification (Silla and Freitas, 2011; Zhang and Zhou, 2014), and can be viewed as an extension of this direction onto sequences tagging.

8 Conclusions

We proposed a tag-hierarchy model for the heterogeneous tag-sets NER setting, which does not require a consolidation post-processing stage. In the conducted experiments, the proposed model consistently outperformed the baselines in difficult tagging cases and showed robustness when applying a single trained model to varied test sets.

In the case of integrating datasets from the news and medical domains we found the blending task to be difficult. In future work, we'd like to improve this integration in order to gain from training on examples from different domains for tags like '*Name*' and '*Location*'.

Acknowledgments

The authors would like to thank Yossi Matias, Katherine Chou, Greg Corrado, Avinatan Hassidim, Rony Amira, Itay Laish and Amit Markel for their help in creating this work.

References

Hector Martinez Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *EACL 2017-15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1–10.

Isabelle Augenstein, Sebastian Ruder, and Anders Søgaard. 2018. Multi-task learning of pairwise sequence classification tasks over disparate label spaces. *arXiv:1802.09913v2*.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *ACL*.

Johannes Bjerva. 2017. Will my auxiliary tagging task help? estimating auxiliary tasks effectivity in multi-task learning. In *Proceedings of the 21st Nordic Conference on Computational Linguistics, NoDaLiDa, 22-24 May 2017, Gothenburg, Sweden*, 131, pages 216–220. Linköping University Electronic Press.

Hongshen Chen, Yue Zhang, and Qun Liu. 2016. Neural network for heterogeneous annotations. In *EMNLP*.

Jason Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *TACL*, 4(1):357–370.

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

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12(Aug):2493–2537.

Franck Dernoncourt, Ji Young Lee, Ozlem Uzuner, and Peter Szolovits. 2017. De-identification of patient notes with recurrent neural networks. *J. Am Med Inform Assoc*, 24(3):596–606.

Chuanhai Dong, Huijia Wu, Jiajun Zhang, and Chengqing Zong. 2017. Multichannel lstm-crf for named entity recognition in chinese social media. In *CCL/NLP-NABD*. Springer.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. 2000. Physiobank, physiotoolkit, and physionet. *Circulation*, 101(23):215–220.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP*.

Nathan Greenberg, Trapit Bansal, Patrick Verga, and Andrew McCallum. 2018. Marginal likelihood training of bilstm-crf for biomedical named entity recognition from disjoint label sets. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2824–2829.

Kazuma Hashimoto, Yoshimasa Tsuruoka, Richard Socher, et al. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *EMNLP*.

Hangfeng He and Xu Sun. 2017. A unified model for cross-domain and semi-supervised named entity recognition in chinese social media. In *AAAI*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv:1508.01991*.

Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. 2015. New transfer learning techniques for disparate label sets. In *ACL*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *ACL*.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2018. Transfer learning for named-entity recognition with neural networks.

Zhenghua Li, Jiayuan Chao, Min Zhang, Wenliang Chen, Meishan Zhang, Guohong Fu, Zhenghua Li, Jiayuan Chao, Min Zhang, Wenliang Chen, et al. 2017. Coupled pos tagging on heterogeneous annotations. *TASLP*, 25(3):557–571.

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramon Fernandez, Silvio Amir, Luis Marujo, and Tiago Luis. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *EMNLP*.

Zengjian Liu, Buzhou Tang, Xiaolong Wang, and Qingcai Chen. 2017. De-identification of clinical notes via recurrent neural network and conditional random field. *J. Biomed. Inf.*, 75:34–42.

Rune B Lyngsø and Christian NS Pedersen. 2002. The consensus string problem and the complexity of comparing hidden markov models. *Journal of Computer and System Sciences*, 65(3):545–569.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Nanyun Peng and Mark Dredze. 2017. Multi-task domain adaptation for sequence tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Lizhen Qu, Gabriela Ferraro, Liyuan Zhou, Weiwei Hou, and Timothy Baldwin. 2016. Named entity recognition for novel types by transfer learning. In *EMNLP*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv:1706.05098*.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Benjamin Shickel, Patrick James Tighe, Azra Bihorac, and Parisa Rashidi. 2017. Deep ehr: A survey of recent advances in deep learning techniques for electronic health record (ehr) analysis. *IEEE Journal of Biomedical and Health Informatics*.

Carlos N Silla and Alex A Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

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

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *ENNLP*.

Amber Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *J. Biomed. Inf.*, 58:20–29.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *NAACL*.

Özlem Uzuner, Yuan Luo, and Peter Szolovits. 2007. Evaluating the state-of-the-art in automatic de-identification. *J. Am Med Inform Assoc*, 14(5):550–563.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. Ontonotes release 5.0 ldc2013t19. *LDC*.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83.

Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837.

A Experiment Results

Full experiment results for Section 6.1

F1			Model		
Tag	Base	Extending	**Hier**	**Indep**	**MTL**
Date	***I2B2'14***	***I2B2'06***	0.899	0.904	0.903
		Onto	0.713	0.686	0.671
	I2B2'06	***I2B2'14***	0.871	0.840	0.875
		Onto	0.641	0.681	0.698
	Onto	***I2B2'14***	0.837	0.830	0.831
		I2B2'06	0.834	0.826	0.807
Hospital	***I2B2'14***	***I2B2'06***	0.931	0.941	0.918
	I2B2'06	***I2B2'14***	0.867	0.866	0.853
Location	***Conll***	***I2B2'14***	0.818	0.783	0.812
		I2B2'06	0.748	0.739	0.730
		Onto	0.836	0.830	0.836
	I2B2'14	***Conll***	0.954	0.899	0.887
		I2B2'06	0.953	0.919	0.919
		Onto	0.951	0.921	0.907
	I2B2'06	***Conll***	0.876	0.816	0.760
		I2B2'14	0.886	0.883	0.888
		Onto	0.869	0.847	0.812
	Onto	***Conll***	0.747	0.701	0.703
		I2B2'14	0.793	0.691	0.707
		I2B2'06	0.814	0.691	0.666
Name	***Conll***	***I2B2'14***	0.855	0.771	0.690
		I2B2'06	0.827	0.666	0.631
		Onto	0.860	0.841	0.867
	I2B2'14	***Conll***	0.900	0.863	0.890
		I2B2'06	0.943	0.893	0.927
		Onto	0.911	0.882	0.891
	I2B2'06	***Conll***	0.662	0.679	0.653
		I2B2'14	0.834	0.824	0.808
		Onto	0.726	0.726	0.727
	Onto	***Conll***	0.895	0.888	0.890
		I2B2'14	0.892	0.872	0.886
		I2B2'06	0.846	0.827	0.809