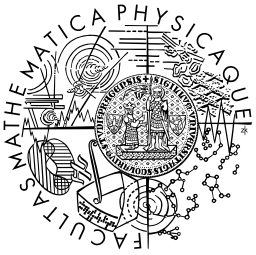


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

**MASTER THESIS**

Mostafa Abdou

# An investigation of the task of Universal Semantic Tagging using Neural Architectures, Multi-task Learning, and Multi-lingual Learning

Institute of Formal and Applied Linguistics

| | |
|---|---|
| Supervisor of the master thesis: | Mgr. Barbora Vidová Hladká |
| Study programme: | European Masters in Language and Communication Technologies |
| Study branch: | Mathematical Linguistic |

Prague 2018

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In 14/07/2018 signature of the author

Dedication.

This thesis would not have been possible without support from my family and from Vinit, Artur, and Daniel whom I had a great time, worked, and learned a great deal with. I would like to thank my supervisors in Groningen, prof. Johan Bos and dr. Lasha Abzianidze, for leading me towards the topic of this thesis and for their valuable advice and feedback.

Title: An investigation of the task of Universal Semantic Tagging using Neural Architectures, Multi-task Learning, and Multi-lingual Learning

Author: Mostafa Abdou

Institute: Institute of Formal and Applied Linguistics

Supervisor: Mgr. Barbora Vidová Hladká, Institute of Formal and Applied Linguistics

Abstract: In this thesis we present an investigation of multi-task and transfer learning using the recently introduced task of semantic tagging. First we employ a number of natural language processing tasks as auxiliaries for semantic tagging. Secondly, going in the other direction, we employ semantic tagging as an auxiliary task for three different NLP tasks: Part-of-Speech Tagging, Universal Dependency parsing, and Natural Language Inference. We compare full neural network sharing, partial neural network sharing, and what we term the *learning what to share* setting where negative transfer between tasks is less likely. Finally, we investigate multi-lingual learning framed as a special case of multi-task learning. Our findings show considerable improvements for most experiments, demonstrating a variety of cases where multi-task and transfer learning methods are beneficial.

Abstrakt: V diplomové práci prezentujeme výzkum paralelního a přenosového učení s využitím nedávno představené úlohy sémantického značkování. Zaprvé vybrané úlohy počítačového zpracování přirozeného jazyka používáme jako podpůrné úlohy pro sémantické značkování. Zadruhé se vydáváme opačným směrem, a sice sémantické značkování používáme jako podpůrnou úlohu pro tři různé úlohy počítačového zpracování přirozeného jazyka: tvaroslovné značkování, parsing na platformě Universal Dependencies a odvozování v přirozeném jazyce. Porovnáváme úplné a částečné sdílení neuronových sítí spolu s učením s méně pravděpodobným nastavením negativního přenosu mezi úlohami. Na závěr zkoumáme vícejazyčné učení v paralelním učení. V experimentech demonstrujeme různé kombinace paralelního učení a přenosového učení. Výsledky jsou pozitivní.

Keywords: Semantic Tagging, Multi-task Learning, Deep Learning, Transfer Learning, Universal Dependencies, Natural Language Inference

Klíčová slova: Sémantické značkování, Paralelní učení, Hluboké učení, Přenosové učení, Univerzální závislosti, dedukce přirozeného jazyka

# Contents

# Introduction

When learning a new skill, humans have a remarkable ability to make use of previously learnt tasks. This can be observed both for closely related tasks where there is considerable similarity between the tasks and for more distantly related tasks where the relationship between the tasks is not very strong. For instance, when learning a new language which is of the same language family as an already acquired language (e.g. Spanish and Portuguese or Czech and Polish), the lexical and grammatical similarities between the two languages will make the acquisition of the second language easier. This, however, isn't restricted to cases where the relationship between the tasks is clear as in the case described above. More loosely related tasks also have the potential to benefit each other. For instance, when learning to drive, the locomotive principles acquired while learning to ride a bike or skateboard could prove beneficial.

Inspired by this idea, the machine learning community began to look for methods of transferring knowledge from one task to another - an area of research now known as transfer learning. Being able to successfully transfer knowledge between tasks poses the clear benefit of eliminating the need for separately gathering and labelling training data for each task in a set of related tasks. Indeed, in various fields, effective transfer learning has reduced the need for the expensive and labour-intensive process of data-collection for every new task which relates to one or more tasks which training data already exists for.

Recently, Computer Vision has become one of the fields where transfer learning is most commonly and successfully employed. For example, generic feature extractors (or encoders) from deep convolutional network models which are trained on a very large amount of labeled data (such as ImageNET [Deng et al., 2009]) have led to state-of-the-art performance on a wide variety of tasks such as object image classification, scene recognition, fine grained recognition, attribute detection and image retrieval — with no or little further training [Razavian et al., 2014]. An important point to note is that models using these off-the-shelf generic feature extractors actually outperformed the best specifically-trained models in many cases, demonstrating the power of transfer learning. Similarly (but arguably to a lesser extent so far) in the field of Natural Language Processing (NLP) transfer learning has become an essential component of the models used for a wide range of tasks. One method in particular which has become ubiquitous, known as Word Embedding [Collobert and Weston, 2008, Mikolov et al., 2013c, Pennington et al., 2014], makes use of unlabeled textual data to construct word representations that can be used off-the-shelf as a starting point in the training of task-specific models. Promising attempts have also been made to extend this approach to sentence-level tasks [Conneau et al., 2017b].

Examples of task relatedness in NLP are easy to come by. Different syntac-

tic or semantic tagsets will, for instance, highly correlate with each other. As a demonstration, consider the following two example sentences annotated with the Penn Treenbank (PTB) part-of-speech tagset [Marcus et al., 1993] and the Universal Semantic Tagset (semtags) [Bjerva et al., 2016].

- You[PRO] have[NEC] to[NIL] take [EXS] the[DEF] first_step[CON] .[NIL]
- You[PRP] have[VBP] to[TO] take[VB] the[DT] first_step[NN] .[.]
- We[PRO] lost[EPS] the[DEF] game[CON] 3-0[SCO] .[NIL]
- We[PRP] lost[VBD] the[DT] game[NN] 3-0[CD] .[.]

Each of the examples is first shown with semtags (red) and then with PTB tags (blue). The semtags used in these two examples are: PRO for pronouns, NEC for necessity, NIL for words with vacous semantics, EXS for untensed simple, DEF for definite, CON for concept, EPS for past simple, and SCO for score. In these examples, the differences between the tagsets is most evident in the semantic distinction which semtags make between concept nouns (CON) and role nouns (ROL) and in the labeling of 'have' as 'necessity' rather than simply as 'Verb, non-3rd person singular present' and with determiners which the PTB tagset groups together as DT while semtags differentiate between definite (DEF), proximal (PRX), and distal determiners (DST). However, in these examples and in general, there is a one-to-one or a one-to-many correspondence between the tags of the two tagsets, meaning that data labeled using one of the tagsets can be exploited for training models of the other. Given that there is a varying amount of labeled data for different tasks in different languages, it is certainly worthwhile to investigate how best the similarities between them can be exploited.

Another more recent domain of research which is very closely connected to transfer learning and which also aims to exploit the similarities between different tasks is multi-task learning [Caruna, 1993]. Among the main distinctions between the two is that in transfer learning we mainly care about doing well on a target task using information from other tasks while in multi-task learning, the goal is to do well on all available tasks. Another distinction is that a similar amount of labeled data is often available for all tasks in multi-task learning, but not in transfer learning. Recently, multi-task learning using neural network methods has surged in popularity due to the relative ease of implementation and a series of successful results.

This research will aim to explore approaches to multi-task learning and transfer learning using (deep) neural network methods within the context of various sequence labelling and semantic tasks. In particular, it will focus on investigating whether and how the newly proposed task of Universal Semantic Tagging can benefit or benefit from other NLP tasks. We choose to focus on this task because: *i)* assigning semantic labels to words is a simple task when compared to building complex relational semantic structures for e.g. semantic parsing; *ii)* a large supervised dataset is publicly available [Abzianidze et al., 2017], in contrast to other semantic tasks such as word sense disambiguation and lexical similarity; *iii)* the semantic tagging task abstracts over syntactic POS tagsets and can be seen as their semantic analogue aimed at being language-neutral and useful for multi-lingual downstream semantic tasks.

## Chapter Guide

This thesis will be divided into five chapters aiming to answer the following research questions:

- **Research Question 1:** Which sequence labelling tasks, if any, can help with semantic tagging in a multi-task learning setting?
- **Research Question 2:** Can semantic tagging be informative for other sequence labelling tasks? If so, how and under which multi-task settings?
- **Research Question 3:** Can semantic tagging be informative for higher-level semantic tasks such as natural language inference or structured prediction tasks such as dependency parsing in a multi-task learning or transfer learning setting?
- **Research Question 4:** Can multi-lingual approaches which treat multi-lingual learning as a sepcial case of multi-task learning be used to improve semantic tagging accuracy for languages with less or no training data?

### Chapter 1 - Background and Previous Work

The aim of Chapter 1 will be to introduce and motivate the methods and theories which are utilized in this thesis and to highlight the most relevant previous work which serves as a foundation for it.

### Chapter 2 - Multi-task Learning for Sequence Labelling Tasks

In Chapter 2 we will investigate multi-task learning for sequence labelling tasks through the task of semantic tagging. Our investigation will aim to explore which sequence labelling tasks could contribute to and benefit from semantic tagging. Experiments involving different settings of multi-task learning using neural network methods such as hard-parameter sharing or learning to share will be presented.

### Chapter 3 - Multi-task Learning for Sentence-level and Structured Prediction Tasks

In Chapter 3 we will look to expand the scope from multi-task learning for sequence labelling tasks to sentence-level (semantic) tasks and structured prediction tasks.

### Chapter 4 - Multi-lingual Learning as an instance of Multi-task learning

In Chapter 4 we will then switch our focus to multi-lingual learning as a specific case of multi-task learning where each language is treated as a task. Particularly, we will focus on whether we can improve the performance of our semantic tagging models for low-resourced languages through the use of multi-lingual representations.

**Chapter 5 - Conclusions**

Finally, in Chapter 5 we will present our conclusions with respect to each of the research questions and offer future directions of work.

## Published work

This thesis draws from the following work by its author:

- Abdou, M., Ravishankar, V., Kulmizev, A., Abzianidze, A., Bos, J. (Under Review). "What can we learn from Semantic Tagging?" Proceedings of the 12th International Workshop on Semantic Evaluation (EMNLP 2018)
- AffecThor at SemEval-2018 Task 1: A cross-linguistic approach to sentiment intensity quantification in tweets M Abdou, A Kulmizev, JG i Ametllé Proceedings of The 12th International Workshop on Semantic Evaluation, 210-217

All experiments in this thesis have been performed by the author.

## Overview of Experiments

| RQ | Target Task | MTL | TL | MTL Settings | Multi-lingual learning |
|---|---|---|---|---|---|
| RQ1 | Semantic Tagging | Yes | No | Fully-shared networks | No |
| RQ2 | POS tagging | Yes | No | Fully-shared networks<br>Partially-shared networks<br>*Learning what to share* | No |
| RQ3 | Dependency Parsing<br>Natural Language Inference | Yes | Yes | Fully-shared networks<br>Partially-shared networks<br>*Learning what to share* | No |
| RQ4 | Semantic Tagging | Yes | Yes | Fully-shared networks | Yes |

Table 1: Overview of experiments run in this thesis.

# Chapter 1

# Background and Previous Work

## 1.1 Distributional Semantic models and Word Embeddings

Popularized by Firth [1957] as *you shall know a word by the company it keeps*, the distributional hypothesis states that words with similar distributional properties (i.e. which occur in similar contexts) should have similar semantic properties. This approach to language modeling was first empirically utilized in Information Retrieval systems which employed measures such as document word frequency and word co-occurrence counts. Modern NLP has become heavily reliant upon this approach in which a word is typically represented by a high-dimensional vector which captures its co-occurrence statistics in a corpus, thus mapping the original linguistic problem into a geometrical space. Figure 1.1 shows a toy example of a three-dimensional vector-space built with a three word vocabulary. As can be seen, related words are closer together (in terms of angular distance) in the vector space.

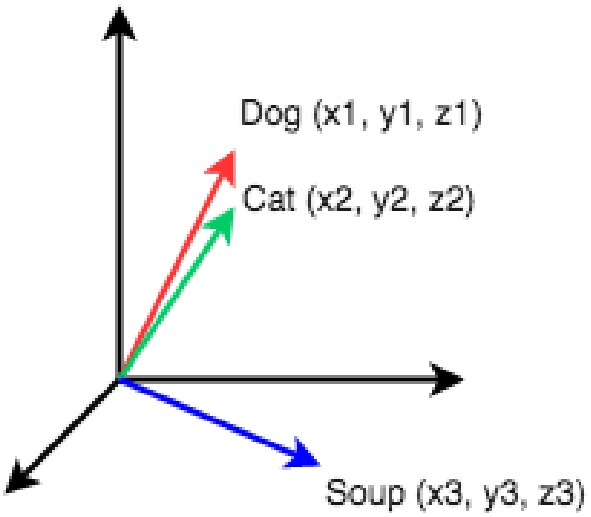



Figure 1.1: A three-dimensional vector-space model.

These geometric *vector space models* have been shown to capture various linguistic notions such as *semantic relatedness* (e.g. teacher being related to student)

and *semantic similarity* (e.g. teacher ≈ professor) among other linguistic relations. An example of this can be seen in Figure **??**, shows a toy example of how these vector space models can capture semantic and syntactic analogy relationships through simple linear relationships, as first demonstrated by Mikolov et al. [2013c].

An important distinction is often made between the traditional sparse context-counting models [Turney and Pantel, 2010, Erk, 2012] and the dense context-predicting models [Mikolov et al., 2013a, Collobert et al., 2011, Huang et al., 2012] in which the vector weights are directly set to predict the contexts in which the corresponding words tend to appear . The latter, better known as *word embeddings*, have recently garnered much attention in the NLP community, and have shown state-of-the-art performance in many tasks [Baroni et al., 2014].

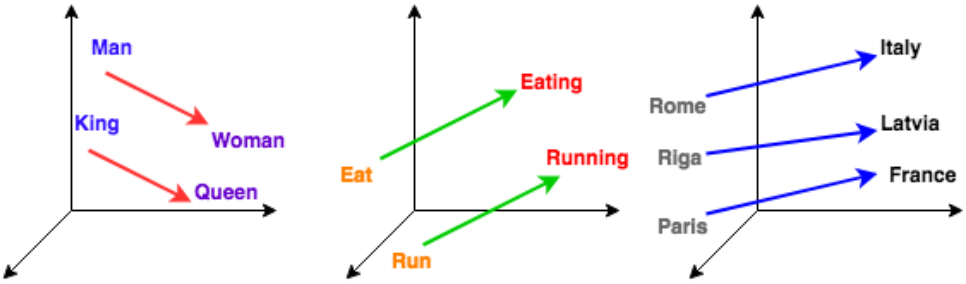



Figure 1.2: Semantic and syntactic analogies are captured by vector space models as linear relationships.

Current state-of-the-art word embedding methods obtain rich word representations either by learning to predict the contexts in which words tend to appear [Mikolov et al., 2013a] or by leveraging global co-occurrence statistics and matrix factorization techniques [Pennington et al., 2014]. Among the most commonly used algorithms to accomplish this are:

- **Skip-gram:** Given a word $w$, predict the surrounding words in context window $c$. (*Context-prediction model*)
- **Continuous bag-of-words:** Predict the current target word $w$ based on the words in a context window $c$. (*Context-prediction model*)
- **GloVe :** Computes ratios of word co-occurrence probabilities (rather than their co-occurrence probabilities themselves). In order to accomplish this, GloVe uses a weighted least squares objective which aims to reduce the difference between the dot product of the vectors of two words and the logarithm of their co-occurrence count. (*Co-occurrence counts and matrix factorization model*)
- **FastText :** Introduces an extension of the Skip-gram model by learning representations for sub-word units (character n-grams), and representing words as the sum of the sub-word units vectors. [Bojanowski et al., 2016] (*Context-prediction model*)

Notable work by Levy and Goldberg [2014] demonstrated the equivalence of the now ubiquitous Word2Vec Skip-gram with negative sampling embeddings to

implicitly factorizing a word-context matrix whose cells are the pointwise mutual information of the respective word and context pairs, shifted by a global constant. This provided an important link to previous work which is based on matrix factorization and dimensionality reduction techniques.

In this thesis, pre-trained word embeddings are used to initialize our models - namely, to provide an initial word-level or token-level representation - as has become standard in a multitude of modern NLP tasks. This can be seen as a form of transfer learning whereby information which is learnt through unsupervised training on a large amounts of text is transferred to other 'downstream' tasks.

## Multi-lingual Word Embeddings

Soon after the emergence of the efficient word embedding methods, described in the previous section, various ideas aiming at extending these methods into the realm of multi-lingual representations were proposed. This was motivated by the need for a way to transfer knowledge in cross-lingual applications. In particular, the availability of high quality multi-lingual word representations would enable transfer of knowledge between resource-rich and low-resource languages, possibly leading to significant improvements for various NLP tasks on the latter.

In Chapter 4, we will investigate the use of multi-lingual representations for transfer of knowledge from well-resourced languages with a large amount of training data to languages with a much smaller amount of training data for the task of semantic tagging.

There is a rich history of work on constructing multi-lingual vector-space models, however, we will restrict ourselves to a brief overview which focuses mainly on the methods which we will be making use of in this work - namely, word-level alignment methods. These methods are the most prevalent and the most successful in the literature. They include:

- **Mapping of monolingual embedding spaces:** In these methods, separate monolingual corpora are trained and then a transformation matrix which maps between the representations of the two spaces is learnt. Mikolov et al. [2013b]'s observation that geometric relationships between words are similar across languages gave rise to the idea of learning a linear projection from one space to the other that minimizes the sum of squared Euclidean distances between pairs of words in bi-lingual dictionary of seed words. Dinu et al. [2014] improve this method by altering the objective function to prevent words from clustering too closely to each other in *hubs*. Later work by Xing et al. [2015], Zhang et al. [2016], Artetxe et al. [2016] employs length normalization (to unit length) and maximizes cosine similarity, adding an orthogonality constraint for the transformation matrix to preserve the length normalization after projection. The trend has been towards needing less and less bi-lingual dictionary seed item until, finally, Conneau et al. [2017c] propose a completely unsupervised method based on a discriminative adversarial objective. This is the class of methods which we will be using in this thesis.

- **Synthetic bilingual corpora based methods:** Instead of learning a mapping between two monolingual spaces, this class of methods uses a seed

dictionary to build a bilingual corpus by randomly replacing words in a monolingual corpus with their translations.

- **Joint learning methods:** Instead of learning separate monolingual embedding spaces and mapping them, this class of methods aims to directly learn multi-lingual embedding spaces by taking in parallel text as input and minimizing the monolingual losses jointly with a multi-lingual regularization term. This has also been, so far, a less common approach and we will not make use of it in this thesis.

## 1.2 Neural Methods in Natural Language Processing

Since Bengio et al. [2003]'s seminal work of neural language modeling and subsequently the emergence of the word embeddings methods described in Section 1.1, it has become more and more common to for Natural Language Processing (NLP) practitioners to utilize a family of machine algorithms known as neural networks (NNs), particularly those of the 'deep' variety. As this family of methods is at the core of this thesis, we will present an overview of it in this section. We start with brief history and proceed to describe some of the most common variants, especially those which are widely used for NLP tasks and which we will make use of.

### 1.2.1 History and Resurgence

In the 1930s and 1940s several researchers, such as Warren McCulloch, Walter Pitts, and Donald O. Hebb, first began working on algorithms and computational models which can vaguely be seen as drawing inspiration from humans' neural machinery. In 1958, Rosenblatt [1958] proposed and implemented the Perceptron, one of the first built NNs. Research into NNs, however, soon enough stagnated considerably due to two factors: *i)* Research revealing various deficiencies such as the inability of a Perceptron to model the exclusive-or function [Minsky et al., 2017] and *ii)* The lack of sufficient computational processing power to support networks which were large enough to accomplish significant tasks.

Rapid advances in hardware and the introduction of the efficient training regime known as Backpropagation [Werbos, 1974] which enabled the training of networks with multiple layers lead to a revival in interest in NNs, which then burst back onto the scene with work like Hinton et al. [2006], Bengio et al. [2003] in the early 2000s preceding a startling domination of machine learning fields, including computer vision, bioinformatics, and NLP.

### 1.2.2 Neural Methods in NLP

One of the main problems with trying to model language problems (or other problems with discrete random variables) using neural methods is that the sequences of words (variables) seen during training is likely to be different from the sequences seen at testing time. This problem which is due to inherent productive nature of language, is referred to as *the curse of dimensionality* by Bengio

et al. [2003]. For example, when trying to model the joint probability of a sentence made up of 20 consecutive words drawn from a vocabulary of $50,000$ words, there are $50,000^{20} - 1$ possible free parameters. Instead, the authors demonstrate that modeling words as continuous variables by learning a distributed representation for each word allows for a far better degree of generalization.

This idea only really took hold a few years later with Collobert and Weston [2008] devising an general neural model which can jointly learn to perform many NLP tasks (part-of-speech tagging, chunking, named-entity recognition, semantic role-labelling, and language modelling) at once. From there on, the flood gates opened, with neural methods achieving state-of-the-art performance on many NLP tasks in the following few years. This included: pos-tagging [Huang et al., 2015], parsing [Chen and Manning, 2014], language modelling [Mikolov et al., 2010, Kim et al., 2016], and machine translation [Sutskever et al., 2014, Bahdanau et al., 2014, Vaswani et al., 2017b].

### 1.2.3 Common Neural Architectures

In this subsection we will go over a few of the most commonly utilized neural network methods, providing insight into the theoretical underpinnings of each of the methods which we will later use.

#### Feed-forward Networks

A Feed-forward network (FFN), also known as a multilayer perceptron (MLP), is one of the earliest and simplest neural architectures. FFNs are typically used in supervised learning problems. It is made up of a network of neurons[1] called Perceptrons [Rosenblatt, 1958]. Perceptrons compute an output from multiple inputs as a linear combination according to a set of weights that is learned, then pass the output through an activation function (often a non-linearity). A single Perceptron is of modest utility because the class of functions it can compute is limited. No matter what activation function is used, the Perceptron is unable to learn a problem with a non-linear decision boundary, such as for instance, the exclusive-or function [Minsky et al., 2017]. However, if a single so called hidden-layer of neurons is added between the input layer and the output layer, modeling such functions becomes possible. In fact, it has been shown that with enough hidden units, a single-layer[2] FFN with an arbitrary squashing function [3] is a *universal function approximator*, i.e. it can approximate any function from one finite dimensional space to another[4], no matter how complex [Hornik et al., 1989]. This is a fascinating idea, because it means that using what is essentially a series of matrix-vector multiplications with non-linearities applied to them, we can approximate a function which maps from any given space to another. This is remarkable because essentially any problem you can think of can be formulated as function computation. Take for instance the problem of translating from French

[1] 'Neuron' is the common nomenclature, however, we would like to emphasize that it has become well recognized that the link to biological neurons is tenuous if at all existent.

[2] single hidden-layer

[3] e.g.: A sigmoidal function

[4] caveat: this only applies to continuous functions

text into English. This can be thought of as the computation of one of many[5] functions which map correctly from a sequence of French words to a sequence of English words. If we can adequately approximate one of those functions then we would be able to correctly translate the sequence. Of course, universality theorems like this do not in any way guarantee that we will be to build the network which is able to approximate a given function, only that it exists.

Figure 1.3 shows a single hidden-layer NN. Let us now formalize our definition: An FFN is a neural network which consist of an input layer (the first layer from the left), one or more hidden layers (the middle layer), and an output layer (the rightmost layer) where a layer consist of $N$ neurons (the white circles) each layer is fully-connected[6] (these connections are represented by arrows) to the next with weighted connections. Mathematically, each layer can be represented as a vector of activations and each set of weights between layers can be represented as a matrix. As such, the network in Figure 1.3 can be written as:

$$\vec{y} = \sigma(\mathbf{W_1}(\sigma(\mathbf{W_0}\vec{x} + \vec{b_0})) + \vec{b_1}) \tag{1.1}$$

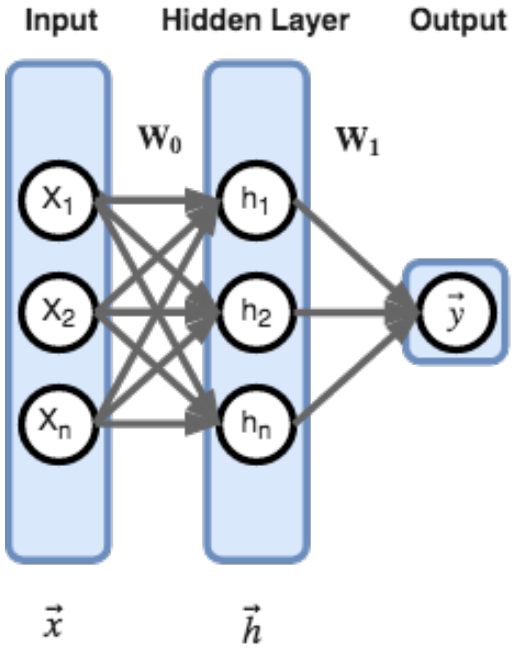



Figure 1.3: A simple Feed-forward Network (bias vectors and activations not shown).

It's worth taking a moment out to comment on the notation used in Equation 1.1 as we will be using the same notation for the rest of this thesis. Each layer is denoted as a vector ($\vec{y}$, $\vec{h}$, and $\vec{x}$) and each set of weights between two layers as a matrix ($\mathbf{W_0}$, $\mathbf{W_1}$). Activation functions are represented as $\sigma$. $\vec{b}$ denotes a bias vector with the subscript indicating the layer it belongs to. Bias allows the network to shift the activation function, which has proven critical for successful learning.

[5]since there are often many possible correct translations of a natural language sequence.
[6]i.e. each neuron in a layer is connected to every neuron in the next layer

### 1.2.4 Convolutional Networks

Convolutionals Neural Networks (CNN) are a neural architecture loosely inspired by the visual mechanism of humans. Specifically, CNNs take inspiration from the visual cortex which has regions of cells which are sensitive to particular regions of the visual field. The current version of CNNs was proposed by LeCun et al. [1995] and has led to immense improvements in the field of computer vision, and more recently, in NLP. There are many possible variations of this architecture, but a typical CNN consists of three types of layers, namely convolutional, pooling, and fully-connected layers. The first is composed of several convolution kernels which are used to compute different feature maps. The complete feature maps are obtained by using several different kernels. The pooling layer then aims to achieve generalization (shift-invariance) by reducing the resolution of the feature maps. After a few convolutional and pooling layers, there is normally one or more feedforward layers which are supposed to perform high-level operations. This is then followed by an output layer which is chosen dependent on the task.

A simple CNN can be seen in Figure 1.4. The input is an image[7], i.e. a matrix of size $m \times n \times r$ where $m$ is height, $n$ is width, and $r$ is the number of channels[8]. This is followed by a convolutional layer where $k$ filters of dimension $h \times w \times c$, where $h \times w < m \times n$ (height x width), and $c \leq r$ (channels), are slid over the image. This produces a representation of the image that is based on $w \times h$ sized slices of the input image, i.e. a blurred or scaled-down representation. Each filter shares the same weights with the units to which it is connected. This weight-sharing has been crucial to the success of CNNs, greatly reducing the number of parameters compared to a Feed-forward layer, and endowing CNNs with their shift-invariance. Each filter produces a feature map which extracts the same type of features (e.g. edges, curves, etc.) from the input in various locations. Each feature map is then sub-sampled through mean or max pooling [9] in the pooling layer, resulting in condensed feature maps. Finally, this is followed by a Feed-forward layer from which the output is obtained.

Unlike the example shown in Figure 1.4, the process of convolution and pooling is normally repeated more than once. Each convolution and pooling layer shrinks the spatial dimension of the image (i.e. reduces its resolution) and increases its depth (i.e. the number of feature maps). In recent literature, CNNs that are as deep as 152 layers [He et al., 2016] have been used.

CNNs hold clear advantages over Feed-Forward networks, particularly for computer vision applications such as image recognition where information regarding spatial structure is of prime importance. In NLP, where the input is some sort of textual representation (a string of words or characters) rather than images, things differ somewhat. Use of one dimensional CNNs and shallower networks is more common. CNNs in this case can be as something similar to an n-gram feature detector given a window or words or characters.

The use of CNNs has become more and more common in NLP, due to their

[7] Although our focus in this thesis will be on textual data, we chose a demonstrative example based on images because CNNs were first envisioned with visual data in mind and are therefore more readily understood from that viewpoint.

[8] A greyscale image will have just one channel. A RGB image, by contrast, will have three channels, one for each of the colors.

[9] other arithmetic operations can also be used

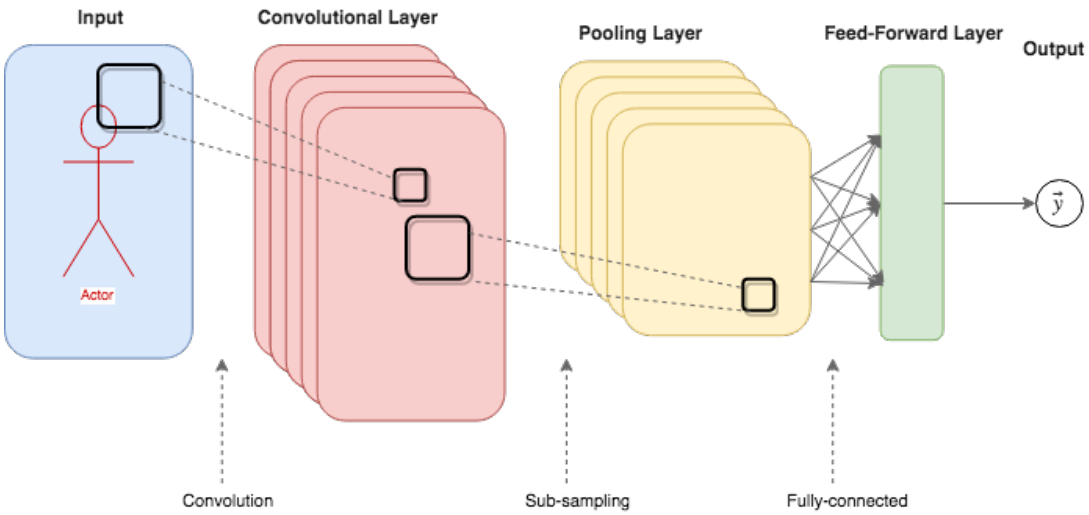



Figure 1.4: A simple Convolutional Neural Network where the input is an image.

requirement of a relatively small number of parameters and the possibility of parallelization (unlike recurrent networks). Examples of this include sentence classification tasks [Kalchbrenner et al., 2014]; convolutional sentence encoders for machine translation [Gehring et al., 2016]; language modelling [Kim et al., 2016]; and sequence labelling tasks such as in Bjerva et al. [2016]'s deep residual tagger, which we employ in this thesis.

### 1.2.5 Recurrent Networks

Recurrent neural networks (RNN) are a class of neural networks which was proposed by Elman [1990] as a means of "finding structure in time". Over the past few years, they have become the tool of choice for many NLP tasks ranging from part-of-speech tagging to language modeling and machine translation. This popularity has been primarily due to their capacity to incorporate the sequential structure of input into account, introducing a notion of time. A RNN is essentially a sequence of FNNs with each FNN connected to the one which follows it in a sequence of adjacent time steps. A RNN takes as input a sequence of arbitrary length $(x_1, x_2, ..., x_t)$ and returns another $(\hat{o_1}, \hat{o_2}, ..., \hat{o_t})$. At a time $t$ a RNN node receives two inputs: $x_t$, the input corresponding to the current time step, and $h_{t-1}$ the hidden node value from the previous time step, producing an output $\hat{o_t}$, based on $h_t$. Therefore, at time step $t$, the output $\hat{o_t}$ has access to information from all previous $x_{t-1}$ inputs by way of the recurrent connections between the nodes, but not to information from future time steps. Figure 1.5 shows an example of this. On the left side of the figure the RNN is represented as FNN with a loop and the right side the unfolded view of an RNN with $t$ time steps is shown.

Formally put, the calculation needed to compute the hidden state $h_t$ of an RNN at time step $t$ can be specified using the following equation:

$$h_t = \sigma(\mathbf{W_x}x + \mathbf{U}h_{t-1} + b_h) \tag{1.2}$$

where $\mathbf{W_x}$ is the weight matrix for the input $x_t$ at time $t$, $\mathbf{U}$ is the weight matrix for the connection from the previous time step and $b_h$ is the bias parameter. The

output $\hat{o}_t$ can then be defined as:

$$\hat{o}_t = \sigma(\mathbf{W_o} h_t + b_o) \tag{1.3}$$

where $\mathbf{W_o}$ is the output weights matrix, and $b_o$ is the bias parameter.

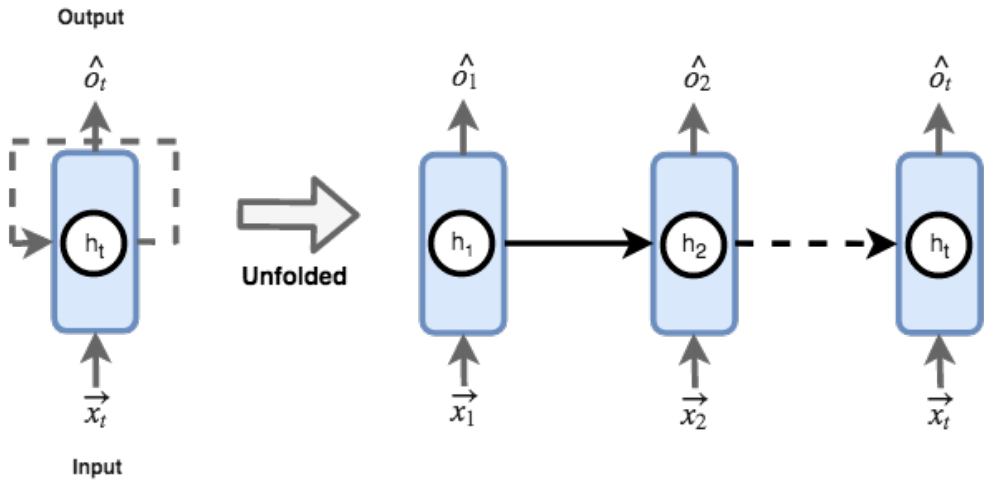



Figure 1.5: A simple recurrent network. On the left side is the representation of the network as a FNN with recursion and on the right side is the unfolded with a connection from each time step to the following time step till time step $t$.

From another viewpoint, RNNs can be seen as recursive neural networks where the recursion takes the structure of a linear chain. The main advantage offered by this recursion is that at a certain time step $t$, the network has access to information from previous time steps. For instance, for the task of word sense disambiguation, given the word *play* as input at time $t$, an RNN would be able to consult the hidden state representation which would indicate that the previous words were either *performed*, *a*, and *Greek*, for instance, disambiguating *play* as a dramatic work, or *children*, *started*, and *to*, disambiguating it as "engage in activity for enjoyment and recreation".

Vanilla RNNs as described so far are, however, rarely used in modern day applications. Although in theory they should be able to capture long-term dependencies between inputs (e.g. between the input at time step $t_1$, $x_1$ and the input at time step $t_{15}$, $x_{15}$), the representations produced by them have been found in practice to be biased towards the most recent inputs. This is due to the way the method in which neural networks are normally trained - backpropagation[10] - works. Specifically, Vanilla RNNs have been known to suffer from two main problems: *vanishing gradients* and its counterpart *exploding gradients*. Both problems result from sharing the weights matrices across many time steps. *Vanishing gradients* can occur due to a large number of multiplicative operations on numbers that are squashed by an activation function into the range $-1 < r < 1$, as in the case of a *tanh* activation function. In this case, the contribution of an input $x_{t-f}$ to the output at time step $t$ will exponentially approach zero as the time step $f$ moves further away (earlier) from $t$, which will mean that the gradient with respect input $x_{t-f}$ will vanish. Intuitively, the opposite, *exploding*

[10]backpropagation through time in the case of RNNs

*gradients*, can also occur when matrices of large weights are multiplied. For a detailed description and a mathematical treatment of the problems faced when training vanilla RNNs, see Pascanu et al. [2013].

Since RNNs only take into account contexts in one direction, it is has become common practice to use Bidirectional RNNs (BI-RNNS) which processes a sequence in both directions (from $x_1$ to $x_t$ and from $x_t$ to $x_t$) and concatenate[11] the outputs produced by both passes [Schuster and Paliwal, 1997].

### Long Short-Term Memory and Gated Rectified Units

To deal with the problem of unstable gradients which makes training RNNs difficult, Hochreiter and Schmidhuber [1997] proposed an enhanced version, the Long Short-Term Memory (LSTM). LSTMs have the same overall structure as RNNs, but are augmented with memory cells. These memory cells replace the nodes in the hidden layer. Each memory cell is equipped with three gates which are designed to modulate the flow of information through the LSTM: an input gate, a forget gate, and an output gate. The gates control the state of memory cell, determining which information should be retained, and which should be discarded. Formally, an LSTM unit can be specified as follows:

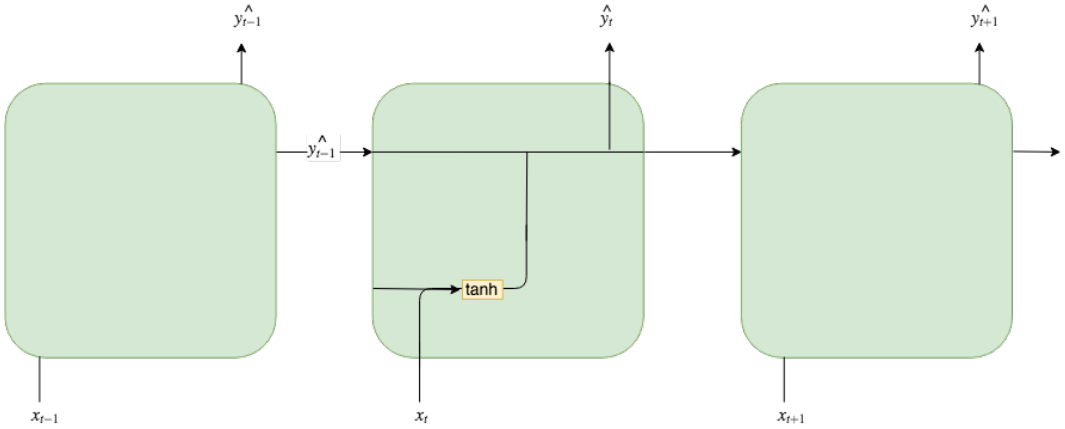
$$
\begin{aligned}
g_t &= tanh(\mathbf{W_g} x_t + \mathbf{U_g} \hat{y}_{t-1} + b_g), \\
i_t &= \sigma(\mathbf{W_i} x_t + \mathbf{U_i} \hat{y}_{t-1} + b_i), \\
f_t &= \sigma(\mathbf{W_f} x_t + \mathbf{U_f} \hat{y}_{t-1} + b_f), \\
o_t &= \sigma(\mathbf{W_o} x_t + \mathbf{U_o} \hat{y}_{t-1} + b_o), \\
s_t &= i_t \odot g_t + f_t \odot s_{t-1}, \\
\hat{y}_t &= o_t \odot \sigma(s_t)
\end{aligned}
\tag{1.4}
$$

where $g_t$ is the input node. The input node takes in the input $x_t$ of the current time step from the input layer and the output from the hidden layer at the previous time step $\hat{y_{t-1}}$. Similarly, the input gate takes in the input $x_t$ of the current time step from the input layer and the output from the hidden layer at the previous time step $\hat{y_{t-1}}$. A gate is used to control information flow by multiplying its value to that of a node. If, for instance, the value is zero, then no information passes through. Whereas, if it's value is one, then all the information flows through. $i_t$ is the output of the input gate. The value of the input gate is multiplied by the value of the input node. $f_t$ is the output of the forget gate. The forget gate is used to enable the network to exclude irrelevant information from the internal cell state, $s_t$, which is at the core of an LSTM unit. $o_t$ is the output of the output gate which is multiplied by the internal cell state $s_t$[12] to obtain $\hat{y}_t$, the output vector. An example LSTM unit can be seen in sub-Figure 1.6b.

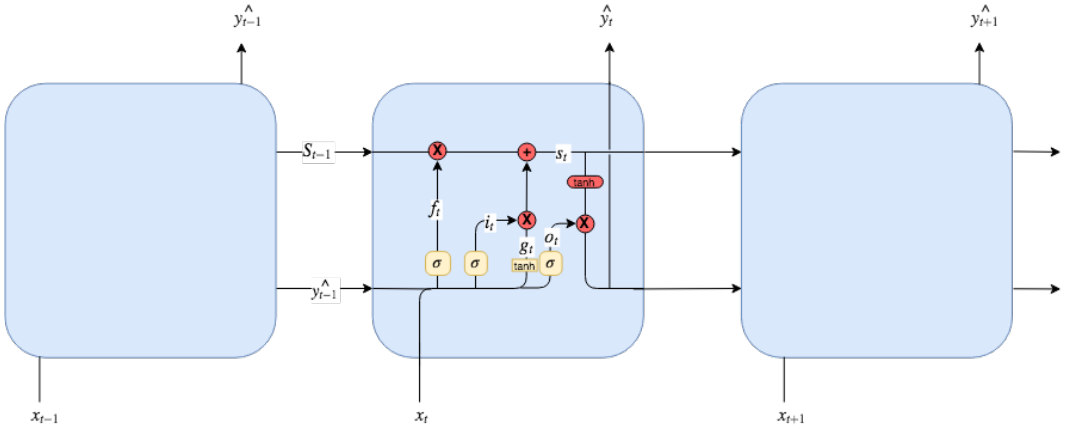
Gated Recurrent Units (GRU) are an another variant of RNNs which was recently proposed by Cho et al. [2014]. GRUs do not make use of an internal memory cell, instead relying only on update and reset gates to modulate the flow of information. This means that GRUs have less parameters and are thus faster

[11] Although concatenation is the most commonly applied operation, it is also possible to perform other operations such as maximum or average pooling of the two vectors.

[12] often with an activation function applied to the internal cell state first.




(a) Vanilla RNN




(b) LSTM

Figure 1.6: Three time steps of a vanilla RNN and an LSTM. The internal breakdown is shown for both at time step $t$. Pointwise operations are indicated in red and neural network layers in yellow. The LSTM's gates and nodes are labelled in white in sub-Figure b.

to train and require less training. In practice, however, GRUs an LSTMs have been found to perform comparably [Chung et al., 2014].

The idea of bidirectionality described in section 1.2.5 is also commonly employed for LSTMS and GRUs. In recent work, models making use of bidirectional LSTMs (BI-LSTM) and bidirectional GRUs (BI-GRU) have been shown to achieve state-of-the-art results on a variety on NLP tasks. Indeed, they have become standard for tasks such as: part-of-speech tagging [Plank et al., 2016], dependency parsing [Dozat and Manning, 2016], and machine translation [Sutskever et al., 2014], among many others. In this thesis, we make extensive use of BI-LSTMs.

## 1.3 Sequence Labelling and Sentence Classification Tasks

A large number of NLP tasks can be put under one of two general classes of tasks: sequence labeling tasks and sentence classification tasks. In the former the objective is to assign a label to each element in a sequence of elements. In the most common scenario, the sequence is a sentence and each element is a word. The labels we assign are often things like part-of-speech tags, named entity labels, etc. Common sequence labelling tasks include:

- Part-of-speech tagging
- Syntactic chucking
- Named Entity Recognition
- Relation Extraction (Between entities)
- Semantic Role Labelling

In sentence classification tasks, on the other hand, we consider one or more sentences at a time with the goal of producing a label for either one sentence, a group of sentence, or the relationship between them. Examples of common sentence classification tasks include:

- Sentiment analysis (binary or fine-grained)
- Customer/Movie/product review classification
- Sentence level semantic similarity classification
- Natural language inference
- Question-type classification
- Paraphrase detection
- Image caption retrieval

Our focus in this thesis will be on the task of Universal Semantic Tagging, a sequence labelling task. The task is described in detail in the next subsection. We will also run experiments with various other sequence labelling tasks such as part-of-speech tagging and sentence classification tasks such as Natural Language Inference.

## Universal Semantic Tagging

Universal semantic tagging [Bjerva et al., 2016, Abzianidze and Bos, 2017a] is the task of assigning language-neutral semantic categories to words. It is designed to overcome a lack of semantic information syntax-oriented part-of-speech tagsets, such as the Penn Treebank tagset [Marcus et al., 1993], usually have. Such tagsets exclude important semantic distinctions, such as negation and modals, types of quantification, named entity types, and the contribution of verbs to tense, aspect, or event.

The semantic tagset is language-neutral, abstracts over part-of-speech and named-entity classes, and includes fine-grained semantic information. The tagset consists of 80 semantic tags grouped in 13 coarse-grained classes. The tagset originated in the Parallel Meaning Bank (PMB) project [Abzianidze et al., 2017], where it contributes to compositional semantics and cross-lingual projection of semantic representations. Recent work has highlighted the utility of the tagset as a conduit for evaluating the semantics captured by vector representations [Belinkov et al., 2018], or employed it in an auxiliary tagging task [Bjerva et al., 2016], as we do in this thesis.

To highlight the differences between PTB tags and semtags we include the examples below:

- You[PRO] have[NEC] to[NIL] take[EXS] the[DEF] first_step[CON] .[NIL]
- You[PRP] have[VBP] to[TO] take[VB] the[DT] first_step[NN] .[.]
- We[PRO] lost[EPS] the[DEF] game[CON] 3-0[SCO] .[NIL]
- We[PRP] lost[VBD] the[DT] game[NN] 3-0[CD] .[.]
- Tom[PER] left[EPS] his[HAS] wife[ROL] a[DIS] fortune[CON] .[NIL]
- Tom[NNP] left[VBD] his[PRP$] wife[NN] a[DT] fortune[NN] .[.]
- My[HAS] son[ROL] is[NOW] playing[EXG] in[REL] the[DEF] rain[CON] .[NIL]
- My[PRP$] son[NN] is[ VBZ] playing[VBG] in[IN] the[DT] rain[NN] .[.]

Each of the examples is first shown with semtags (red) and then with PTB tags (blue). The semtags used in these examples are: PRO for pronouns, NEC for necessity, NIL for no meaning, EXS for untensed simple, DEF for definite, CON for concept, EPS for past simple, SCO for score, PER for person, HAS for

Table 1.1: The Universal Semantic Tagset v0.1.0: 73 semtags (highlighted in blue) grouped into 13 meta-tags (highlighted in red). Examples are included for each semtag and further context is included for more ambiguous semtags. (Adapted from Abzianidze and Bos [2017b].)

**ANA** anaphoric
- PRO anaphoric & deictic pronouns: *he, she, I, him*
- DEF definite: *the, lo$^{IT}$, der$^{DE}$*
- HAS possessive pronoun: *my, her*
- REF reflexive & reciprocal pron.: *herself, each_other*
- EMP emphasizing pronouns: *himself*

**ACT** speech act
- GRE greeting & parting: *hi, bye*
- ITJ interjections, exclamations: *alas, ah*
- HES hesitation: *err*
- QUE interrogative: *who, which, ?*

**ATT** attribute
- QUC concrete quantity: *two, six_million, twice*
- QUV vague quantity: *millions, many, enough*
- COL colour: *red, crimson, light_blue, chestnut_brown*
- IST intersective: *open, vegetarian, quickly*
- SST subsective: *skillful* surgeon, *tall* kid
- PRI privative: *former, fake*
- DEG degree: 2 meters *tall,* 20 years *old*
- INT intensifier: *very, much, too, rather*
- REL relation: *in, on, 's, of, after*
- SCO score: *3-0,* grade *A*

**COM** comparative
- EQU equative: *as* tall as John, whales *are* mammals
- MOR comparative positive: *better, more*
- LES comparative negative: *less, worse*
- TOP superlative positive: *most, mostly*
- BOT superlative negative: *worst, least*
- ORD ordinal: *1st, 3rd, third*

**UNE** unnamed entity
- CON concept: *dog, person*
- ROL role: *student, brother, prof., victim*
- GRP group: John {,} Mary *and* Sam gathered, a *group* of people

**DXS** deixis
- DXP place deixis: *here, this, above*
- DXT temporal deixis: *just, later, tomorrow*
- DXD discourse deixis: *latter, former, above*

**LOG** logical
- ALT alternative & repetitions: *another, different, again*
- XCL exclusive: *only, just*
- NIL empty semantics: {.}, *to, of*
- DIS disjunction & exist. quantif.: *a, some, any, or*
- IMP implication: *if, when, unless*
- AND conjunction & univ. quantif.: *every, and, who, any*

**MOD** modality
- NOT negation: *not, no, neither, without*
- NEC necessity: *must, should, have* to
- POS possibility: *might, could, perhaps, alleged, can*

**DSC** discourse
- SUB subordinate relations: *that, while, because*
- COO coordinate relations: *so,* {,}, {;}, *and*
- APP appositional relations: {,}, *which,* {(}, {—}
- BUT contrast: *but, yet*

**NAM** named entity
- PER person: *Axl_Rose, Sherlock_Holmes*
- GPE geo-political entity: *Paris, Japan*
- GPO geo-political origin: *Parisian, French*
- GEO geographical location: *Alps, Nile*
- ORG organization: *IKEA, EU*
- ART artifact: *iOS_7*
- HAP happening: *Eurovision_2017*
- UOM unit of measurement: *meter, $, %, degree_Celsius*
- CTC contact information: *112, info@mail.com*
- URL URL: `http://pmb.let.rug.nl`
- LIT literal use of names: his name is *John*
- NTH other names: table *1a,* equation *(1)*

**EVE** events
- EXS untensed simple: to *walk,* is *eaten, destruction*
- ENS present simple: we *walk,* he *walks*
- EPS past simple: *ate, went*
- EXG untensed progressive: is *running*
- EXT untensed perfect: has *eaten*

**TNS** tense & aspect
- NOW present tense: *is* skiing, *do* ski, *has* skied, *now*
- PST past tense: *was* baked, *had* gone, *did* go
- FUT future tense: *will, shall*
- PRG progressive: has been *being* treated, *aan_het$^{NL}$*
- PFT perfect: has *been* going/done

**TIM** temporal entity
- DAT ag full date: *27.04.2017, 27/04/17*
- DOM day of month: *27th* December
- YOC year of century: *2017*
- DOW day of week: *Thursday*
- MOY month of year: *April*
- DEC decade: *80s, 1990s*
- CLO clocktime: *8:45_pm, 10_o'clock, noon*

possessive pronoun, ROL for role, DIS for disjunction or existential quantifier, NOW for present tense, EXG for untensed progressives, REL for relation, and DEF for definite. In these examples, the differences between the tagsets is most evident in the semantic distinction which semtags make between concept nouns (CON) and role nouns (ROL) and in the labeling of 'have' as 'necessity' rather than simply as 'Verb, non-3rd person singular present'. Moreover, '3-0' is labeled as 'Score' rather than as 'cardinal number'. Another important distinction is that between the semtags for articles (DEF, DIS) which distinguish different types of articles, as opposed to grouping them under one label as is done in the PTB (DT). The full semantic tagset is shown in Table 1.1.

## 1.4 Multi-task learning

Multi-task learning (MTL) is a recently resurgent approach to machine learning in which multiple tasks are simultaneously learned. It is often motivated through a comparison to humans' ability to utilize the knowledge acquired through one task for another (e.g. a person being able to play the guitar makes it easier for them to learn to play the ukulele). By optimizing the multiple loss functions of related tasks at once, multi-task learning models can achieve superior results to models which are trained on a single task. The goal of Multi-task learning is summarized by Caruana [1998] "MTL improves generalization by leveraging the domain-specific information contained in the training signals of related tasks".

The key benefits of MTL revolve around the intuitive idea that simultaneously learning representations for multiple tasks (e.g. semantic tagging and natural language inference) pushes a model to prefer representations which are generally rich (i.e. suitable for multiple tasks) over those which are task-specific. This can be broken down into the following inter-related points:

a) MTL introduces an useful inductive bias helping a model ignore task-specific noise and thereby generalize better, particularly when a task's data is limited or very noisy.

b) MTL can help a model identify relevant features through additional evidence, enabling it to and assign less importance to irrelevant features.

c) In the cases when there is no overlap between two (or more) tasks' training examples, MTL results in the (implicit) augmentation of the training data - allowing the model to generalize better by averaging the task specific noise patterns.

d) MTL can be viewed as performing regularization, reducing a model's ability to fit random noise (i.e. Rademacher complexity).

In the context of neural networks, multi-task learning can be accomplished through hard parameter sharing (shared layers) or soft parameter sharing (each task has own sub-model/parameters, distance between parameters of the sub-models is regularized in order to encourage similarity). In practice, the former approach has been more common due the relative ease of implementing it and

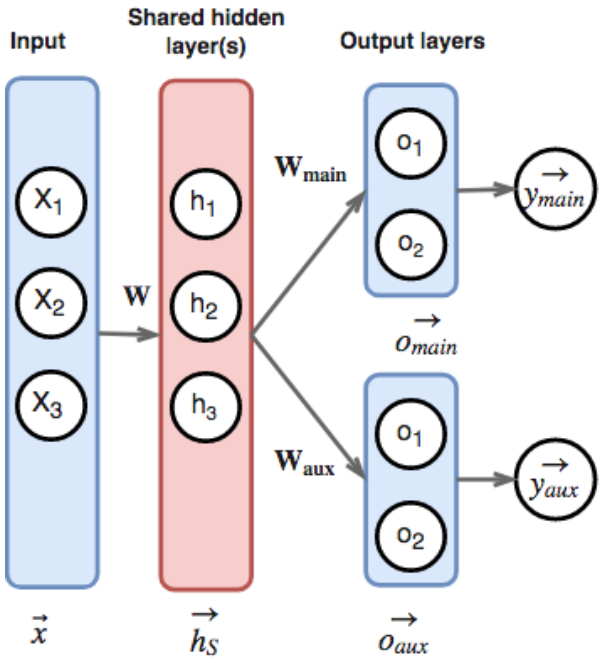



Figure 1.7: A FNN with multi-task learning using full parameter sharing implemented between two tasks: a main task and an auxiliary task.

due to it not requiring an increase in model complexity. An example of this approach can be seen in Figure 1.7. In this example, two tasks, a main task and an auxiliary task share a network's hidden layer $\vec{h_s}$. Each task has its own final output layer. Therefore, the model's parameters are all shared except for the weights from the hidden layer to the two final output layers.

Neural MTL has become an increasingly successful approach for exploiting similarities between Natural Language Processing (NLP) tasks [Collobert and Weston, 2008, Søgaard and Goldberg, 2016, Plank et al., 2016]. While some have aimed to jointly learn multiple standard NLP tasks at once with the aim of improving performance on all tasks [Hashimoto et al., 2016, Bingel and Søgaard, 2017, Collobert and Weston, 2008], others have tried to identify useful auxiliary tasks.

Auxiliary tasks are typically closely related to the main task but not of importance by themselves. Examples of auxiliary tasks used include what Plank [2016b] terms *fortuitous data*. Recent work has been successful at exploiting this kind of data such as Plank [2016a] which uses the signal from keystroke data to enhance a shallow syntactic parsing model and Klerke et al. [2016] which improves sentence compression models by jointly learning to predict gaze.

### Selective sharing

Recently, there has been a particularly interesting direction of research on developing models which are trained to learn what to (and what not to) share between a set of tasks, with the general aim of preventing negative transfer when the tasks are not closely related. With closely related tasks it's possible that fully sharing a network's parameters can help improve the performance of the model for all tasks. However, as the tasks become less and less similar, negative transfer becomes more and more likely due. Taking this into consideration, it becomes clear that full sharing of a network's parameters is not a viable approach to joint learning of distantly related task or for large scale MTL.

For large scale MTL or MTL between distantly related tasks, a mechanism is needed to enable models to learn what to share and what not to share. Several methods of incorporating such selective sharing mechanisms into neural models have been proposed. Liu et al. [2016] present various neural architectures which allow for gated sharing between private (i.e. task-specific) and shared layers. Misra et al. [2016] employ a generalisable approach to MTL using *cross-stitch units* which use learned parameters to optimally combine the private and shared representations from any number of tasks' parallel layers. Extending this approach, Ruder et al. [2017] and Meyerson and Miikkulainen [2017] equip their models with additional learned parameters which mediate the sharing between different sub-spaces (private and shared), different layers, and different skip-connections, allowing for maximal flexibility in sharing.

Promising ideas in the same direction of research have recently also been proposed for reinforcement learning. Teh et al. [2017], for example, propose a "distilled" policy that captures common behavior between tasks. Each individual task's worker is then constrained to stay close to the "distilled" policy while solving its own task.

## 1.5 Transfer learning

As in MTL with an auxiliary task, transfer learning (TL) is a class of methods where the aim is to leverage information from some task(s) for a target task, with the objective of improving performance on the target task. The differences between the two classes of methods is subtle: (i) TL does not assume the availability of training data for the target task enabling zero-shot learning; (ii) tasks are often learnt jointly for MTL whereas in TL pre-training is the common practice.

Figure 1.8 shows a comparison between traditional supervised learning and transfer learning. In transfer learning the amount of task-specific data is normally smaller than the amount of data from related tasks.

TL methods have in the last few years become immensely successful in the field of computer vision [Razavian et al., 2014] where the use of deep convolutional neural networks trained on Imagenet [Deng et al., 2009] as feature extractors has become an industry standard. TL methods have also found success in NLP, albeit to a lesser extent. Since Mikolov et al. [2013a], a seminal work which introduced an efficient method of computing prediction-based word embedding models, it has become more and more common to initialize the first layer of neural NLP models using word embeddings which are learnt on large corpora. While this has

proven useful, particularly in lower-resource scenarios, it has been less clear how perform TL beyond word embedding layer initialization.

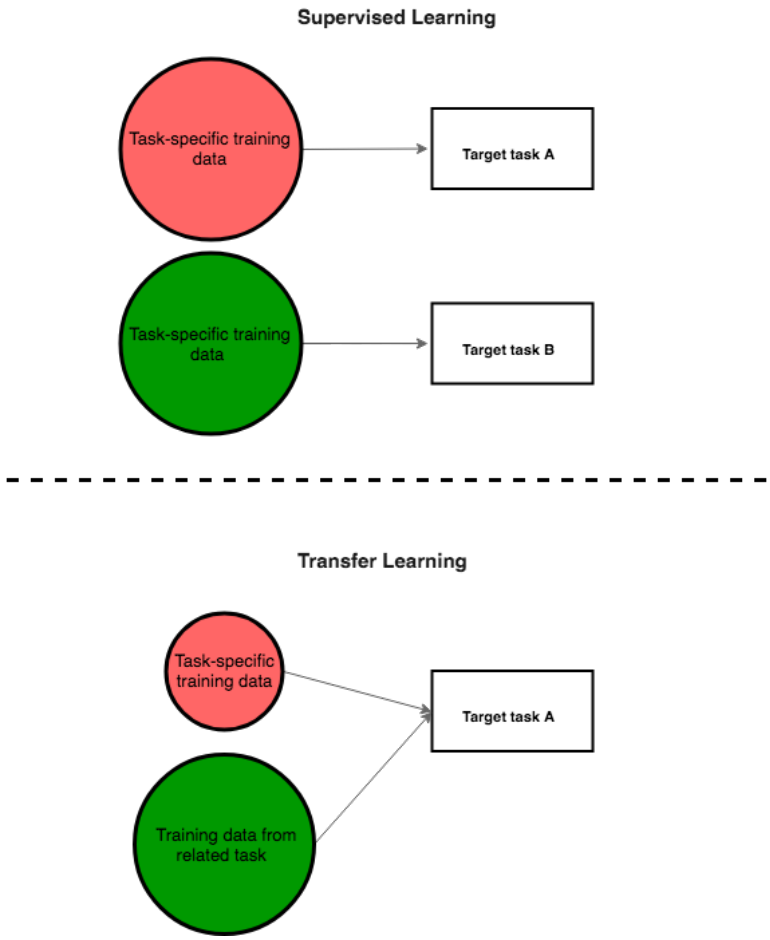



Figure 1.8: A comparison between traditional supervised learning and transfer learning.

In the last year, however, advances have been made in that direction, starting with Conneau et al. [2017a] which found that bi-LSTM with max-pooling sentence encoders trained on SNLI [Bowman et al., 2015b] can perform (i.e. transfer) well on a number of different text classification tasks. Since then, a number of promising works in that direction have emerged: Peters et al. [2018] extract representations from multiple layers of a bidirectional language model which can be fine-tuned on different downstream tasks; Subramanian et al. [2018] use a MTL framework, training on various supervised and unsupervised objectives using the same encoder in order to encode multiple aspects of a sentence; Cer et al. [2018] employ a similar MTL approach, but use a Transformer-based [Vaswani et al., 2017a] encoder rather than a bi-LSTM encoder; and finally, Howard and Ruder [2018] present a very promising approach that closely resembles the TL methods used in computer vision, offering a robust, sample-efficient method for TL using a fixed three layer bi-LSTM language model, and introducing various techniques to enable the fine-tuning on target tasks.

## Cross-lingual Transfer Learning

Finally, we would like to comment on the idea of multi-lingual learning as a special case of MTL or TL, where each language can be seen as a task. Recent work such as Bjerva and Augenstein [2017], Bjerva [2017], Plank et al. [2016], Ammar et al. [2016] has made use of multi-lingual embedding spaces for the purpose of multilingual transfer learning. This is particularly useful for low-resource scenarios where effective model transfer from a well-resourced language (e.g. English) could significantly boost performance. Figure 1.9 shows an example of this scenario, where data from a well-resourced **language B** is used for training a model for the target **language A**, for which there is less training data.

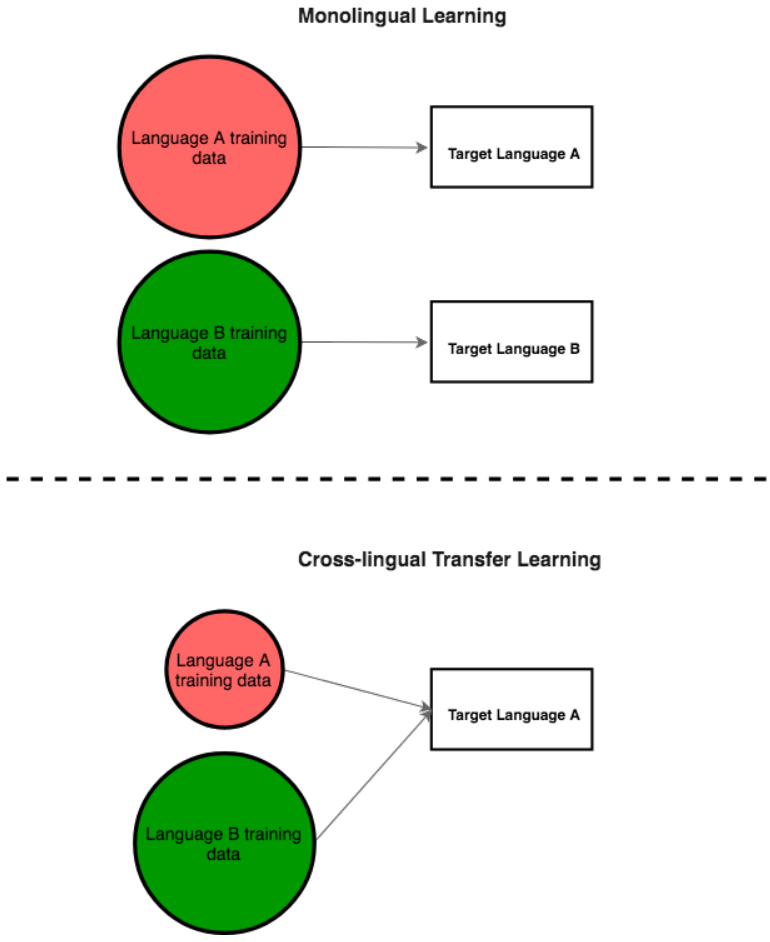



Figure 1.9: A comparison between monolingual learning and cross-lingual transfer learning.

The development of effective unsupervised embedding mapping methods such as Lample et al. [2017] and Artetxe et al. [2018] facilitates the construction of multilingual embedding spaces without the need for bilingual seed dictionaries. However, further work is needed to ascertain when these methods are useful (closely related languages? languages of the same family? all languages?) and when they break.

Recent work has highlighted the various shortcomings of these methods. Nakashole and Flauger [2018] for instance, find that the mappings between the spaces of different languages are not linear. Furthermore, Søgaard et al. [2018] find that

the assumption of (approximate) isomorphism between the word embeddings of different languages (trained on comparable corpora) does not hold up. Additionally, they find that mapping quality significantly deteriorates for linguistically dissimilar languages, proposing a graph eigenvector similarity metric which highly correlates with ability of both supervised and unsupervised methods of mapping.

In this thesis, we will examine the degree to which embedding mapping methods can enable effective multi-lingual learning for semantic tagging.

# Chapter 2

# Multi-task Learning for Sequence Labelling Tasks

In this chapter we will explore the possibilities of mutli-task learning for sequence labelling tasks. In the first section first we will present experiments performed with the goal of improving the performance of a semantic tagging model using a number of other sequence labelling tasks. While doing so, we will also explore the various information theoretic criteria that can be indicative of whether multi-task learning with a certain set of tasks will be successful or not. We will then present an analysis of how well these information theoretic criteria correlate to performance. The goal of this section will be to answer the following research question:

> **Research Question 1:** *Which sequence labelling tasks, if any, can help with semantic tagging in a multi-task learning setting?*

In the second section, we will examine the converse of that question, exploring the space of sequence labelling tasks that can benefit from semantic tagging as an auxiliary task. This section will therefore aim to answer the following research question:

> **Research Question 2:** *Can semantic tagging be informative for other sequence labelling tasks? If so, how and under which multi-task settings?*

## 2.1 Multi-task learning for Semantic Tagging

### 2.1.1 Semantic Tagging

Semantic tagging or *semtagging* is the task of assigning lexical semantic categories to the semantic units of a sentence. The Universal Semantic Tagset we employ for this task is designed to facilitate downstream semantic tasks such as semantic parsing, and is suitable for multi-lingual applications, since the tags are language-neutral. It abstracts over part-of-speech tags and is split into coarse and fine-grained semantic categories. For a detailed look into the tagset, refer to Section 1.3.

#### Semantic Tagging Dataset

The dataset we utilize for our semtagging experiments is derived from the Parallel Meaning Bank (PMB) dataset. We use the latest release of semantic tag data, UST version 0.1.0[1]. This is divided into three parts: gold data which is fully manually corrected, silver data which is partially manually corrected, and bronze data which is automatically labelled using the Trigrams'n'Tags tagger [Brants, 2000]. Table 2.2 shows the number of tokens and sentences in each of the three parts.

| Part | Gold | Silver | Bronze |
|---|---|---|---|
| Sentences | 5568 | 77048 | 176832 |
| Tokens | 35737 | 2443243 | 1829644 |

Table 2.1: The Universal Semantic Tag dataset version 0.1.0's gold, silver, and bronze parts in number of tokens and sentences.

Since the amount of gold and silver data is relatively large, bronze data is excluded from experiments. The entirety of the gold data is used for testing, and the silver data is split into training and development data. The composition of the train, development, and test sets is shown in table 2.2.

| Split | Test | Train | Development |
|---|---|---|---|
| Sentences | 5568 | 75927 | 1121 |
| Tokens | 35737 | 2403038 | 40185 |

Table 2.2: The test, train, and development splits in number of tokens and sentences.

The dataset is tokenised using the Elephant tokeniser Evang et al. [2013], a statistical tokeniser which performs word, multi-word expression, and sentence segmentation. The distributions of the twenty most frequent semtags in the train, development, and testing sets are shown in Figures 2.1a, 2.1b, and 2.1c. We observe that the distribution of labels in the test set differs to that of the train and development sets. This phenomenon, known in machine learning terminology as

[1] http://pmb.let.rug.nl/data.php

Dataset shift, occurs in this case because our silver standard training data is only partially manually corrected while out gold standard test data is fully manually corrected.

### 2.1.2 Method

In order to answer the first research question, we run a set of experiments where semantic tagging is the main task with a variety of sequence labeling tasks as auxiliary tasks. We take two single-task (i.e. without multi-task learning) baselines for semantic tagging: (i) the Stanford log-linear tagger [Toutanova and Manning, 2000], a widely used maximum entropy tagger; (ii) Bjerva et al. [2016]'s deep residual tagger which will be described in detail in sub-section 2.1.4. Both are trained on the full training set. For the latter model, we also experiment with various settings, architecture modifications, and combinations of features.

The auxiliary tasks we consider are:

a) Universal Dependencies Part-Of-Speech Tagging (UPOS)

b) Combinatory Categorial Grammar supertagging (CCG)

c) Universal Dependencies dependency relation tagging (DEPREL)

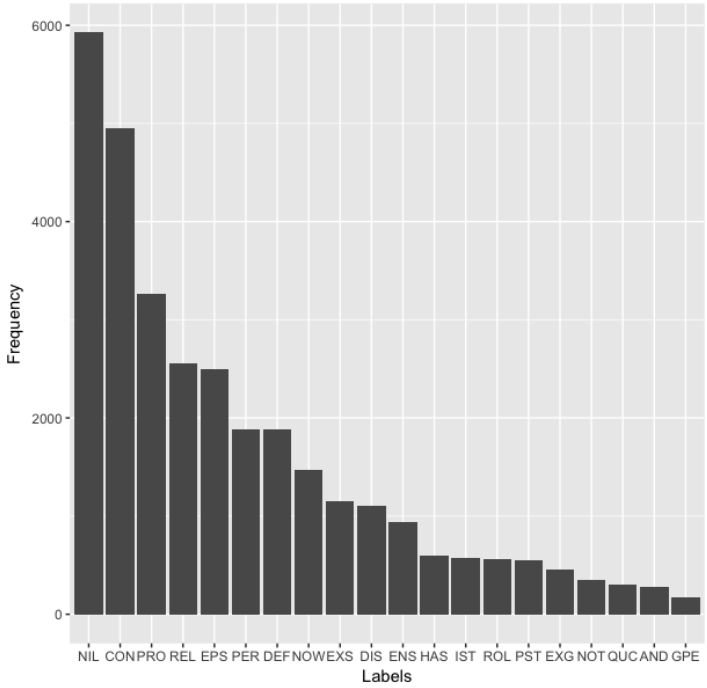
UPOS is a standard part-of-speech tagging task based on the UD part-of-speech tagset. CCG supertagging is the task of assigning CCG lexical categories to the words in a sentence. It is known as supertagging because the labels being assigned are detailed syntactic structures. For this reason, it is sometimes also known as 'almost parsing'. It is, thus, a task which is more difficult than simple part-of-speech tagging. Finally, DEPREL is the task of labelling tokens with their dependency relation labels (i.e. the relations on the edges of the arc pointing to a word in the directed dependency tree).

For each of the three tasks we add an additional classifier to predict the auxiliary task's labels and jointly optimize for both the original semantic tagging task and the auxiliary task. Following Bjerva et al. [2016], we employ hard parameter sharing for MTL in this set of experiments.

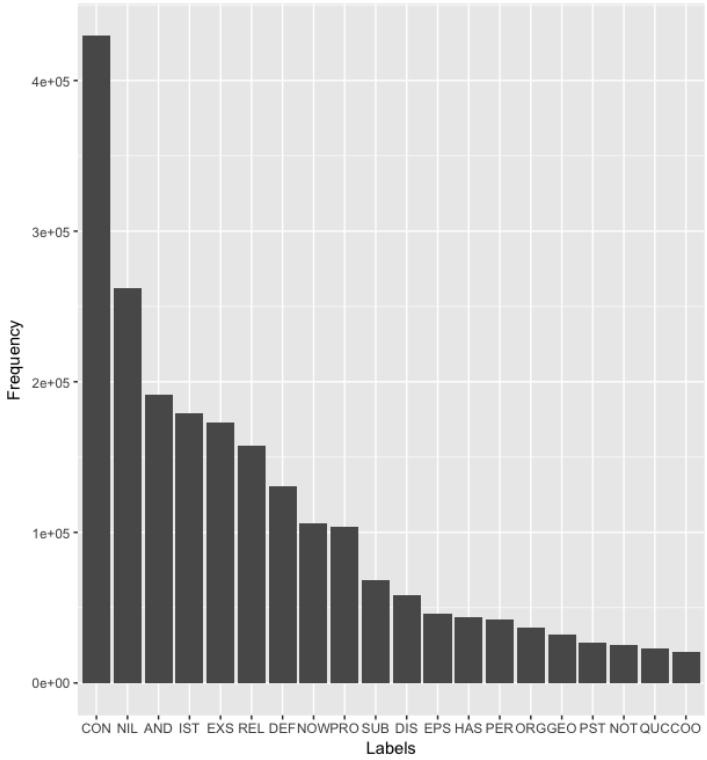
### 2.1.3 Data and Preprocessing

For CCG supertagging the CCG Linguistic Data Consortium distribution [Hockenmaier and Steedman, 2007] is utilized. Following previous work on CCG supertagging [Clark and Curran, 2004], a frequency cutoff of ten occurrences is applied for the categories seen in the training set. Sections 2-21 are used for training, section 00 for development, and section 23 for testing. This is done due to the original number of labels being too large with a considerable number of labels occuring only a few times.

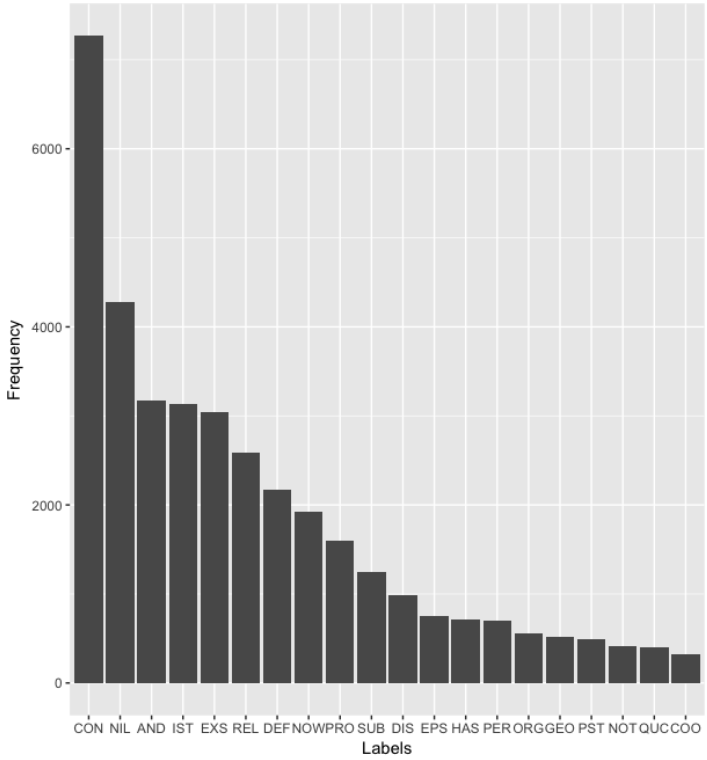
For UPOS and DEPREL, the English data from UD 2.0 [Nivre et al., 2017] is used. The standard splits are maintained for both. For all three datasets the gold tokenisation is used. Table 2.3 shows the training, test, and development splits of the datasets.




(a) Test set distribution




(b) Train set distribution




(c) Development set distribution

Figure 2.1: The frequency distributions of the twenty most frequent semtags in the test, train, and development sets.

| Dataset | Split | Test | Train | Development |
|---|---|---|---|---|
| CCG | Sentences | 4814 | 79208 | 3826 |
| | Tokens | 53077 | 876885 | 43248 |
| UD | Sentences | 6547 | 38169 | 6324 |
| | Tokens | 25097 | 204605 | 25148 |

Table 2.3: The test, train, and development splits in number of tokens and sentences.

### 2.1.4 System

The tagging model we employ is Bjerva et al. [2016]'s deep residual tagger. It is a tagger which utilizes both character and word features through a combination of recurrent and convolutional neural networks with residual connections (ResNets). The tagger is built around a stacked bidirectional LSTM which takes in a combination or word and character features which are composed using either a ResNet or a highway network. At time step $t$, the LSTM's corresponding hidden state $h_t$ is passed on to a dense layer with a softmax activation in order to predict a tag for the word $w_t$. The word features are word embeddings which are initialized with pre-trained GloVe embeddings trained on six billion tokens of common crawl [Pennington et al., 2014] and fine-tuned during training. The character representations are obtained using ResNets or Highway networks with a

base unit consisting of a convolutional neural network followed by a max pooling operation to capture local features around each character of a word. Figure 2.2 shows an overview of the tagger. $\vec{w}$ is the word representation from pre-trained embeddings, $\vec{c}$ is the word representation composed by our model using either a ResNet or a highway network over character embeddings. As can be observed in Figure 2.2, the LSTM is fully shared between the main task (semantic tagging in this case) and the auxiliary task. We set the depth of both ResNets and Highway networks to three.

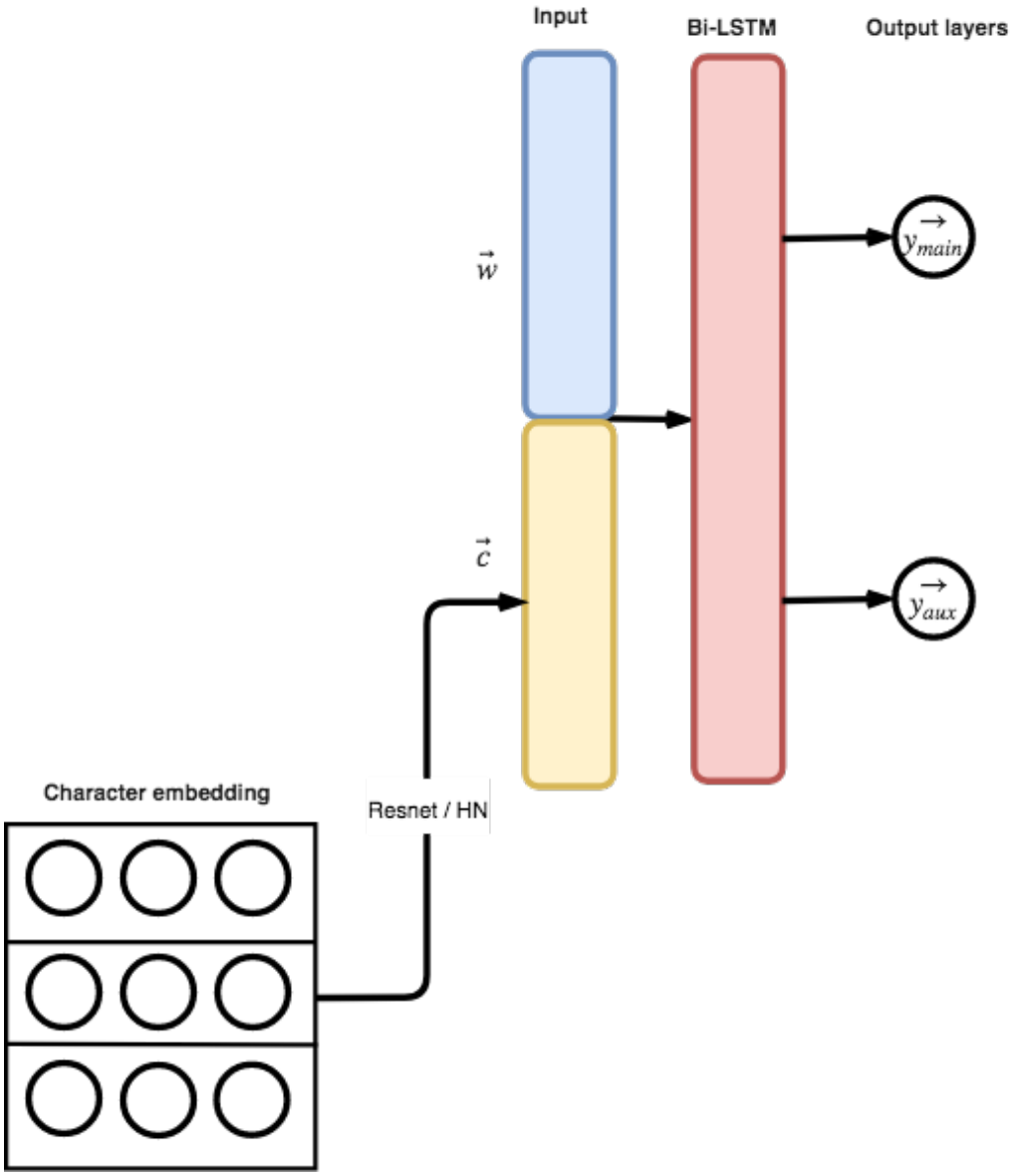



Figure 2.2: A deep residual or highway network tagger.

### Residual Networks

Residual Networks (ResNets) are a recent class of neural models which were developed for computer vision applications. A ResNet unit consists of one or more layers of convolutions with a skip connection short-cutting them. Following

Bjerva et al. [2016], we employ the asymmetric variant of residual units. This can be expressed as follows:

$$x_{l+1} = x_l + F(\sigma(x_l)) \tag{2.1}$$

where $x_l$ is layer $x$ before activation, and $F$ is a convolutional function. This effectively serves as connection between a layer $x_{l+1}$ and the pre-activation $x_l$ of layer $x_l$ by adding that to the output of the activated convolutional block. Figure 2.3 shows an example of residual unit. The CNN or FNN can be replaced with other neural architectures.

The intuition behind ResNets is often explained in terms of them allowing information to flow more easily through the network. This becomes particularly important when the networks are deeper, as it is well-documented that [He et al., 2016] that deeper networks face convergence difficulties. Adding a identity skip connection creates a direct path for error propagation which allows for the efficient training of deep networks. In NLP, ResNets have been successfully used for a variety of tasks, including morphological re-inflection Östling [2016] and text classification [Conneau et al., 2017d].

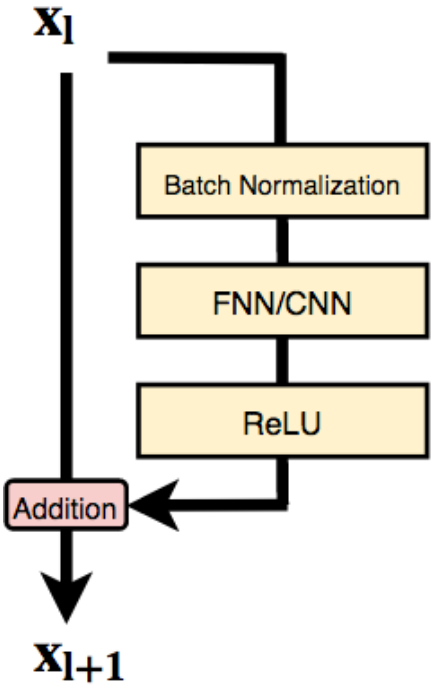



Figure 2.3: A residual unit.

### Highway Networks

Highway networks are an intuitive extension to ResNets which was proposed by Srivastava et al. [2015]. Highway networks augment the skip connections utilized in ResNets with a learnable parameter to determine the extent to which the skip

connections will be utilized. This can be seen in below:

$$x_{l+1} = x_l * (1 - T(x_l)) + T(x_l) * F(x_l) \tag{2.2}$$

In this equation we can see that $T$ the *transform gate* and $1-T$, the *carry gate* are used to determine whether the input is transformed or carried (i.e. passed through as is).

**Hyperparameters, Optimization, and Initialization**

The system's hyperparameters were all tuned with respect to the semantic tagging development set. They are listed below:

- Batch Size: 128
- Epochs: 20
- Optimizer: Adam [Kingma and Ba, 2014]
- Learning rate: 0.0001
- Embedding initialization: Pre-trained GloVe embeddings of dimension 100 trained on 6 billion tokens of Wikipedia 2014 and Gigaword 5
- Dropout: with a probability of 0.3 to all bi-LSTM and embedding layers.
- Auxiliary task weighing: 0.1

### 2.1.5 Results

Table 2.4 shows the results of all experiments using both ResNets and Highway Networks. Semtagging is the baseline model which does not use an auxiliary task. + indicates that a task was used as an auxiliary task.

| **Model** | **Semtagging** | **+UPOS** | **+CCG** | **+DEPREL** |
|---|---|---|---|---|
| Stanford Tagger | 88.81 | NA | NA | NA |
| ResNet | 89.42 | 89.13 | 88.89 | 89.21 |
| Highway Network | 89.29 | **89.55** | 86.08 | 89.23 |

Table 2.4: Results for the Stanford Log-linear Tagger and for our models which employ a ResNet and those which employ a Highway Network. + indicates an auxiliary task. All scores are reported as accuracy.

### 2.1.6 Analysis

The results from this set of experiments indicates that none of the three tasks yield a considerable improvement when employed as an auxiliary for semantic tagging when fully-shared networks are used. Indeed CCG supertagging results in a sizable deterioration in accuracy. The combination of Highway Networks

and UPOS as an auxiliary does however yield the highest accuracy out of all systems. Our findings confirm some of the findings of Alonso and Plank [2016] about when MTL works for semantic sequence prediction tasks. Specifically, i) lower level tasks are less likely to help when employed as an auxiliary for higher level tasks[2]; ii) the main task model will not benefit from an auxiliary loss if it has a large number of labels (as is the case in CCG) and entropy is too high. Indeed, our best performing model uses UPOS, which has a low kurtosis and a compact label distribution, as an auxiliary as is the case in Alonso and Plank [2016]'s only experiment which yields positive results.

## 2.2 Multi-task learning with semantic-tagging as an auxiliary task

In this section we explore the converse of the direction which was explored in the previous section: sequence labelling with semantic tagging as an auxiliary task. We take the task of Universal Dependency POS tagging as the main task and employ semantic tagging as an auxiliary. Instead of expanding our exploration of the space of tasks which could benefit from semantic tagging, we focus on investigating different multi-task settings. We add two selective sharing (see Section 1.4) settings to the multi-task learning setting of fully-shared networks which was employed in the previous section.

### 2.2.1 Learning What to Share

There has recently been an increase in interest in the development of models which are trained to learn what to (and what not to) share between a set of tasks. This is done with the general aim of preventing negative transfer when the tasks are not closely related [Meyerson and Miikkulainen, 2017, Ruder et al., 2017, Lu et al., 2017, Misra et al., 2016]. Our *Learning What to Share* setting belongs to this class of models. It is closely related to Liu et al. [2016]'s shared layer architecture.

Specifically, a layer $\vec{h}_X$ which is shared between the main task and the auxiliary task is split into two subspaces: a shared subspace $\vec{h}_{X_S}$ and a private subspace $\vec{h}_{X_P}$. The interaction between the shared subspaces is modulated via a sigmoidal gating unit applied to a set of learned weights, as seen in Equations (2.3) and (2.4) where $\vec{h}_{X_{S(main)}}$ and $\vec{h}_{X_{S(aux)}}$ are the main and auxiliary tasks' shared layers, $W_{a\to m}$ and $W_{m\to a}$ are learned weights, and $\sigma$ is a sigmoidal function.

$$\vec{h}_{X_{S(main)}} = \vec{h}_{X_{S(main)}}\sigma(\vec{h}_{X_{S(aux)}}W_{a\to m}) \tag{2.3}$$

$$\vec{h}_{X_{S(aux)}} = \vec{h}_{X_{S(aux)}}\sigma(\vec{h}_{X_{S(main)}}W_{m\to a}) \tag{2.4}$$

Unlike Liu et al. [2016], in our setup, each task has its own shared subspace rather than one common shared layer. This enables the sharing of different parameters in each direction (i.e., from main to auxiliary task and from auxiliary to main task).

[2]We take syntactic tasks as being lower-level and semantic tasks as being higher-level in this argument.

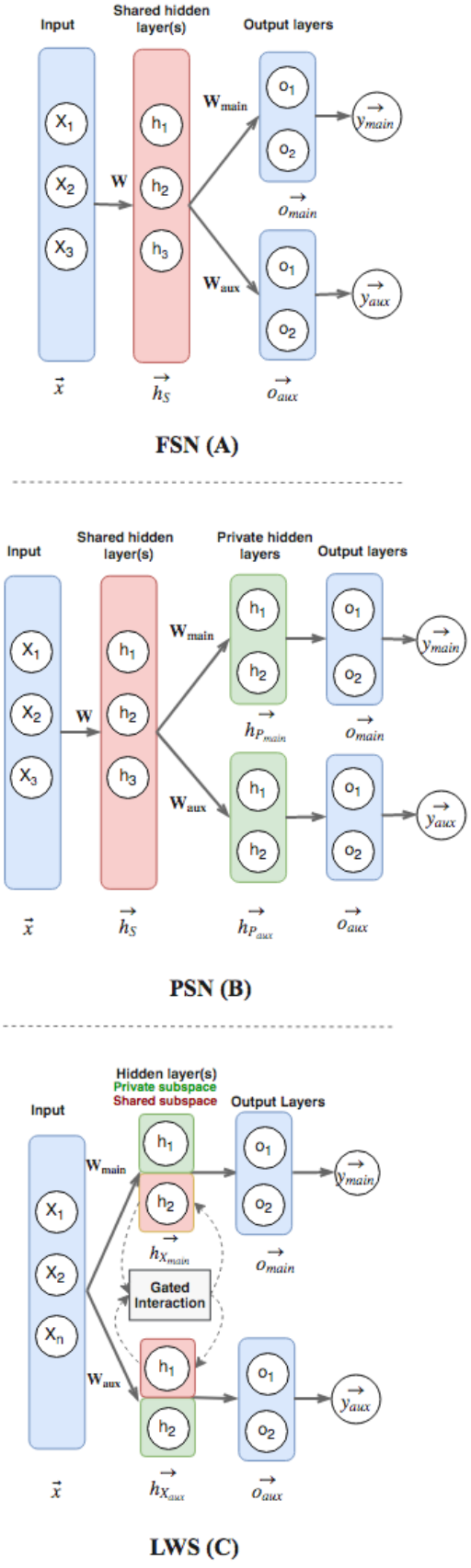



Figure 2.4: The three settings of Multi-task Learning: (A) Fully shared networks, (B) Partially shared networks, and (C) *Learning What to Share*. Layers are mathematically denoted by vectors and the connections between them, represented by arrows, are mathematically denoted by matrices of weights. $S$ indicates a shared layer, $P$ a private layer, and $X$ a layer with shared and private subspaces.

### 2.2.2 Multi-task Learning Settings

We implement three neural MTL settings, shown in Figure 2.4. They differ in the way the network's parameters are shared between the tasks:

- **Fully shared network (FSN):** All hidden layers are entirely shared among the tasks, each task has a separate output layer;
- **Partially shared network (PSN):** A subset of hidden layers is shared among the tasks; each task has at least one private hidden layer and a separate output layer;
- **Learning What to Share (LWS):** Each task has a dedicated set of hidden layers. For sharing, a hidden layer is split into a shared subspace and a private subspace. A gating unit modulates the transfer of information between the shared subspaces as shown in Equations (2.3) and (2.4).

### 2.2.3 Data

We use the datasets described in Sections 2.1.3 and 2.1.1 for Universal Dependencies part-of-Speech tagging (UPOS) and Semantic tagging respectively. Note that there is no overlap between the two datasets, i.e. they do not share the same training examples.

### 2.2.4 Method

In order to answer the second research question, we run a set of experiments where UPOS is the main task and semantic tagging is an auxiliary. We take a single-task (i.e. without multi-task learning) baseline model for UPOS tagging using a bi-LSTM. We then run experiments using the three MTL settings described in Section 2.2.2 to measure the effect of MTL with semantic tagging as an auxiliary task on the UPOS task.

### 2.2.5 System

Our tagging model in this set of experiments uses a basic contextual one-layer bi-LSTM (see Section 1.2.5) that takes in word embeddings and produces a sequence of recurrent states which can be viewed as contextualized representations. The recurrent $r_n$ state from the bi-LSTM corresponding to each time-step $t_n$ is passed through a dense layer with a softmax activation to predict the token's tag. This simple neural architecture has become standard for a number of sequence labelling tasks in the past few years. We choose to not include character features, as our focus is on quantifying the effect of MTL and our word feature model already performs at a level which is close to the state-of-the-art.

In each of the MTL settings a softmax classifier is added to predict a token's semantic tag and the model is then jointly trained on the concatenation of the sem-PMB and UPOS tagging data to minimize the sum of softmax cross-entropy losses of both the main (UPOS tagging) and auxiliary (semantic tagging) tasks. The different MTL setting of the model can be seen in Figure 2.5. In the FSN setting, the hidden layers (i.e. the embedding layer and bi-LSTM) are fully shared. In the PSN setting, the hidden layers are partially shared: the embedding

layer and bi-LSTM are shared then they are followed by a private dense layer for each of the tasks. In the LWS setting, each task has a dedicated set of hidden layers (bi-LSTM). For sharing, the bi-LSTM is split into a shared subspace and a private subspace. A gating unit modulates the transfer of information between the shared subspaces as shown in Equations (2.3) and (2.4).

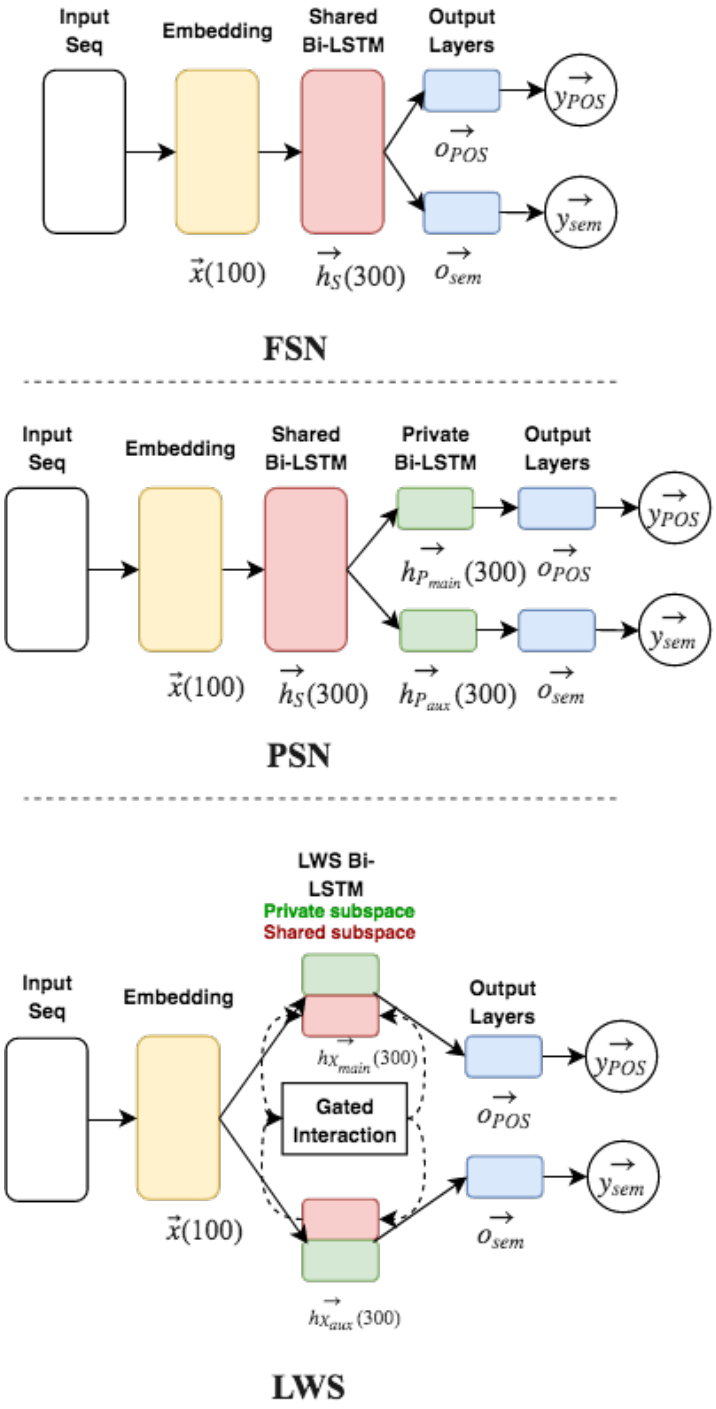



Figure 2.5: The three MTL settings for the tagger. Layers dimensions are displayed in brackets.

**Hyperparameters, Optimization, and Initialization**

All models' hyperparameters were tuned with respect to loss on the English UD 2.0 UD validation set. They are listed below:

- Batch Size: 50
- Epochs: 15
- Optimizer: Adam
- Learning rate: $2e-3$
- Embedding initialization: Pre-trained GloVe embeddings of dimension 100 trained on 6 billion tokens of Wikipedia 2014 and Gigaword 5
- Dropout: with a probability of 0.33 to all bi-LSTM, embedding layers, and non-output dense layers
- Auxiliary task weighing: 0.1

### 2.2.6 Results

Table 2.5 shows results for all MTL setting and the baseline single-task model which does not use an auxiliary task.

| Model | UPOS |
|---|---|
| Single-task | 92.12 |
| Fully-shared Network (FSN) | 92.95 |
| Partially-shared Network (PSN) | 92.34 |
| Learning What to Share (LWS) | **95.54** |

Table 2.5: Results for single-task models (ST), fully-shared networks (FSN), partially-shared networks (PSN), and *Learning What to Share* (LWS). All scores are reported as accuracy.

### 2.2.7 Analysis

In line with Bjerva et al. [2016]'s findings, the FSN setting leads to an improvement for UPOS tagging when semantic tagging is employed as an auxiliary. Our model is simpler than Bjerva et al. [2016]'s in that it does not utilize character features, however the improvement in accuracy still holds. Our findings indicate that while partially-shared networks do not lead to an improvement over the fully-shared network setting, the *Learning What to Share* setting does lead to a considerable improvement. This suggests that adding task-specific layers after fully-shared ones does not always enable sufficient task specialization, whereas employing a selective sharing mechanism does allow for positive transfer between tasks.

It's important to note that all MTL settings outperform the single-task system. We hypothesize that this is down to two main factors: i) the closeness of

the tasks to each other which means that there are useful correlations between the labels of both tasks which the model can learn (e.g. a word tagged as noun in UPOS will often have a unnamed entity semantic tag); ii) the implicit data augmentation which occurs because there is no overlap between the two datasets in terms of training examples. By learning correlations between the tagsets and then training on examples from both tasks, the model is effectively training on more data for both tasks.

# Chapter 3

# Multi-task Learning for Sentence-level and Structured Prediction Tasks

In this chapter we continue with the exploration of multi-task learning with semantic tagging as an auxiliary task. Instead of sequence labelling tasks, we move on to more complex sentence-level and structured prediction tasks. We take the tasks of Universal Dependency parsing and natural language inference as main tasks and employ semantic tagging as an auxiliary. We apply our investigation to all multi-task learning settings which we've introduced so far. Our aim in this chapter is, therefore, to answer the following research question:

> **Research Question 3:** *Can semantic tagging be informative for higher-level semantic tasks such as natural language inference or structured prediction tasks such as dependency parsing in a multi-task learning or transfer learning setting?*

## 3.1 Sentence-level and Structured Prediction Tasks

Sentence-level tasks are a class of NLP tasks which involves assigning labels to sentences as a whole, unlike the sub-sentence labels which are assigned in sequence labelling tasks. This includes tasks like sentiment analysis where a label is normally assigned to a single sentence and it also includes tasks which involve the classification of the relation between two or more sentences such as paraphrase detection or sentence similarity classification. One such task is natural language inference, which involves the classification of the relation between a pair of sentences as either *entailment*, *contradiction*, or *neutral*. This is a task which has recently gained a lot of attention and it is the sentence-level task which we employ in this chapter.

Structured prediction tasks on the other hand involve learning to predict structured outputs rather than discrete labels. The task of dependency parsing is the structured prediction task we consider in this chapter.

As both these tasks are functionally different from semantic tagging which is a sequence labelling task, we expect the flexible selective sharing MTL settings to hold a clear advantage over the full-shared setting in this set of experiments.

### 3.1.1 Universal Dependency Parsing

Universal Dependency parsing (UD DEP) is the task of assigning a syntactic labelled directed graph to a natural language sentence under the following set of constraints:

- There is only one root node that has no incoming arcs.
- Each vertex, except for the root node, has exactly one incoming arc (i.e. no cycles).
- There is one unique path from the root node to each vertex.

These constraints ensure that parses are well-formed trees. An example parse can be seen in Figure 3.1. Dependency parsing can be seen as a task of structured prediciton.

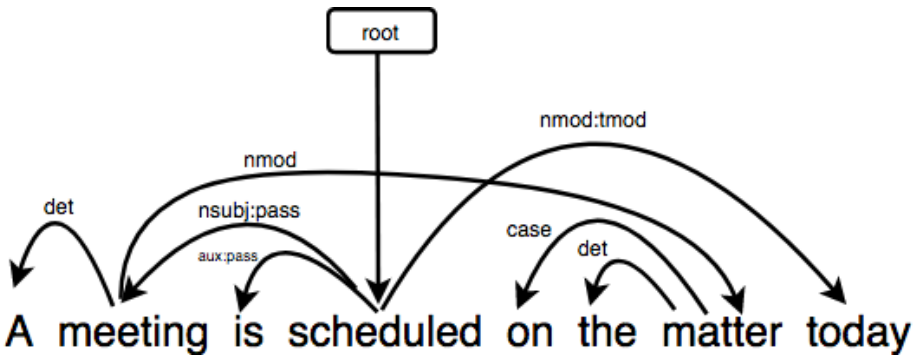



Figure 3.1: An example dependency parsed sentence which is non-projective.

### 3.1.2 Natural Language Inference

Natural language inference (NLI) [1] is the task of recognizing the relation between two sentences (a premise and a hypothesis) as one of either *entailment*, where the first (the premise) sentence entails the second (the hypothesis), *contradiction*, where the second sentence contradicts the first, or *neutral*, where neither applies. Examples of this can be seen in Table 3.1.

| **Premise** | **Hypothesis** | **Classification** |
|---|---|---|
| A black race car starts up in front of a crowd of people. | A man is driving down a lonely road. | Contradiction |
| An older and younger man smiling. | Two men are smiling and laughing at the cats playing on the floor. | Neutral |
| A soccer game with multiple males playing. | Some men are playing a sport. | Entailment |

Table 3.1: Examples of the NLI task taken from the SNLI dataset [Bowman et al., 2015b].

## 3.2 Data and Preprocessing

We use the dataset described in Section 2.1.1 for Semantic tagging. For the UD DEP and NLI experiments there is a complete overlap between the datasets of main and auxiliary tasks, i.e., each instance is labeled with both the main task's labels and semantic tags. We use the Stanford POS Tagger trained on the semantic tagging dataset's training data to tag the UD corpus and NLI datasets with semantic tags, and then use those assigned tags for the MTL settings of our dependency parsing and NLI models.

| **Dataset** | **Split** | **Test** | **Train** | **Development** |
|---|---|---|---|---|
| SNLI | Sentences | 4814 | 79208 | 3826 |
| SICK-E | Sentences | 4927 | 4500 | 500 |
| UD DEP | Sentences | 6547 | 38169 | 6324 |
| | Tokens | 25097 | 204605 | 25148 |

Table 3.2: The test, train, and development splits in number of tokens and sentences.

The UD DEP experiments use the English UD 2.0 corpus, and the NLI experiments use the SNLI [Bowman et al., 2015a] and SICK-E datasets [Marelli et al., 2014]. The provided train, development, and test splits, shown in Table 3.2, are used for all datasets.

[1] also known as recognizing textual entailment (RTE)

## 3.3 Method

In order to answer the third research question, we run four experiments for each of the three tasks (UD DEP, SNLI, SICK-E), one using the single-task (ST) model and one for each of the three MTL settings described in Section 2.2.2 with semantic tagging as an auxiliary task.

## 3.4 Systems

### 3.4.1 Universal Dependency Parsing

We employ a parsing model that is based on Dozat and Manning's [2016] deep biaffine attention dependency parser. The model's embeddings layer is a concatenation of randomly initialized word embeddings[2] and character-based word representations added to pre-trained word embeddings, which are passed through a 4-layer stacked bi-LSTM. Unlike Dozat and Manning [2016], our model jointly learns to perform UPOS tagging and parsing, instead of treating them as separate tasks. Therefore, instead of tag embeddings, we add a softmax classifier to predict UPOS tags after the first bi-LSTM layer. The outputs from that layer and the UPOS softmax prediction vectors are both concatenated to the original embedding layer and passed to the second bi-LSTM layer.

The output of the last bi-LSTM is then used as input for four dense layers with a ReLU activation, producing four vector representations: a word as a dependent seeking its head; a word as a head seeking all its dependents; a word as a dependent deciding on its label; a word as head deciding on the labels of its dependents. These representations are then passed to biaffine and affine softmax classifiers to produce a fully-connected labeled probabilistic dependency graph [Dozat and Manning, 2016]. Finally, a non-projective maximum spanning tree parsing algorithm [Chu, 1965, Edmonds, 1967] is used to obtain a well-formed dependency tree.[3]

Similarly to UPOS tagging, an additional softmax classifier is used to predict a token's semantic tag in each of the MTL settings, as both tasks are jointly learned. In the FSN setting, the 4-layer stacked bi-LSTM is entirely shared. In the PSN setting the semantic tags are predicted from the second layer's hidden states, and the final two layers are devoted to the parsing task. In the LWS setting, the first two layers of the bi-LSTM are split into a private bi-LSTM$_{private}$ and a shared bi-LSTM$_{shared}$ for each of the tasks with the interaction between the shared subspaces being modulated via a gating unit. Then, two bi-LSTM layers that are devoted to parsing only are stacked on top. The models used for each of the three MTL settings are shown in Figure 3.2.

#### Hyperparameters, Optimization, and Initialization

All models' hyperparameters were tuned with respect to loss on the English UD 2.0 UD validation set. They are listed below:

- Batch Size: 50

[2]This replaces the holistic word embeddings for frequent words in Dozat and Manning [2016].
[3]This is recommended but not implemented by Dozat et al. [2017].

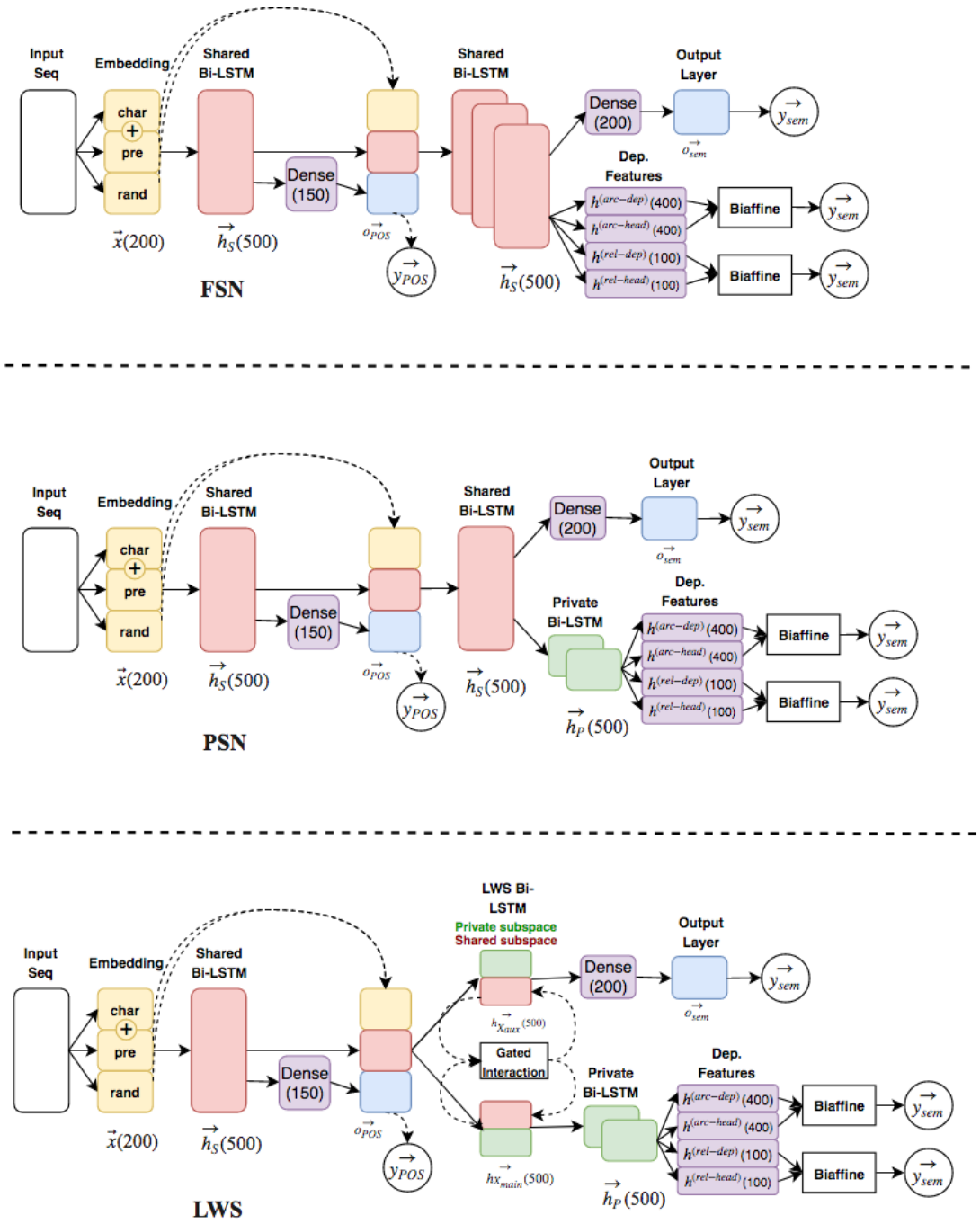



Figure 3.2: The three MTL settings for the UD DEP system. Layers dimensions are displayed in brackets.

- Epochs: 15
- Optimizer: Adam
- Learning rate: $2e-3$
- Embedding initialization: Pre-trained GloVe embeddings of dimension 100 trained on 6 billion tokens of Wikipedia 2014 and Gigaword 5
- Dropout: with a probability of 0.33 to all bi-LSTM, embedding layers, and non-output dense layers

- Auxiliary task weighing: 0.5

### 3.4.2 Natural Language Inference

We base our NLI model on Chen et al. [2017]'s Enhanced Sequential Inference Model which uses a bi-LSTM to encode the the premise and hypothesis, computes a soft-alignment between premise and hypothesis' representations using an attention mechanism, and employs an inference composition bi-LSTM to compose local inference information sequentially.[4] The MTL settings are implemented by adding a softmax classifier to predict semantic tags at the level of the encoding bi-LSTM, with rest of the model unaltered.

In the FSN setting, the hidden states of the encoding bi-LSTM are directly passed as input to the softmax classifier. In the PSN setting an earlier bi-LSTM layer is used to predict the semantic tags and the output from that is passed on to the encoding bi-LSTM which is stacked on top. This follows Hashimoto et al. [2016]'s hierarchical approach. In the LWS setting, a bi-LSTM layer with private and shared subspaces is used for semantic tagging and for the ESIM model's encoding layer. In all MTL settings, the bi-LSTM used for semantic tagging is pre-trained on the semantic tagging training data. he models used for each of the three MTL settings are shown in Figure 3.3.

#### Hyperparameters, Optimization, and Initialization

All models' hyperparameters were tuned with respect to loss on the SNLI validation set for SNLI models and SICK-E validation set fot the SICK-E models. The hyperparameters, optimizer, and initialization settings for both tasks are listed below:

- Batch Size (SNLI): 128
- Batch Size (SICK-E): 8
- Epochs (SNLI): 37
- Epochs (SICK-E): 20
- Optimizer: Adam
- Learning rate: 0.00005
- Embedding initialization: Pre-trained GloVe embeddings of dimension 100 trained on 840 billion tokens common crawl.
- Dropout: with a probability of 0.3 to all bi-LSTM, embedding layers, and non-output dense layers
- Auxiliary task weighing: 0.1

[4]We do not implement the additional tree-LSTM model used in Chen et al. [2017] as we focus on the effect of MTL with semantic tagging rather than on absolute performance.

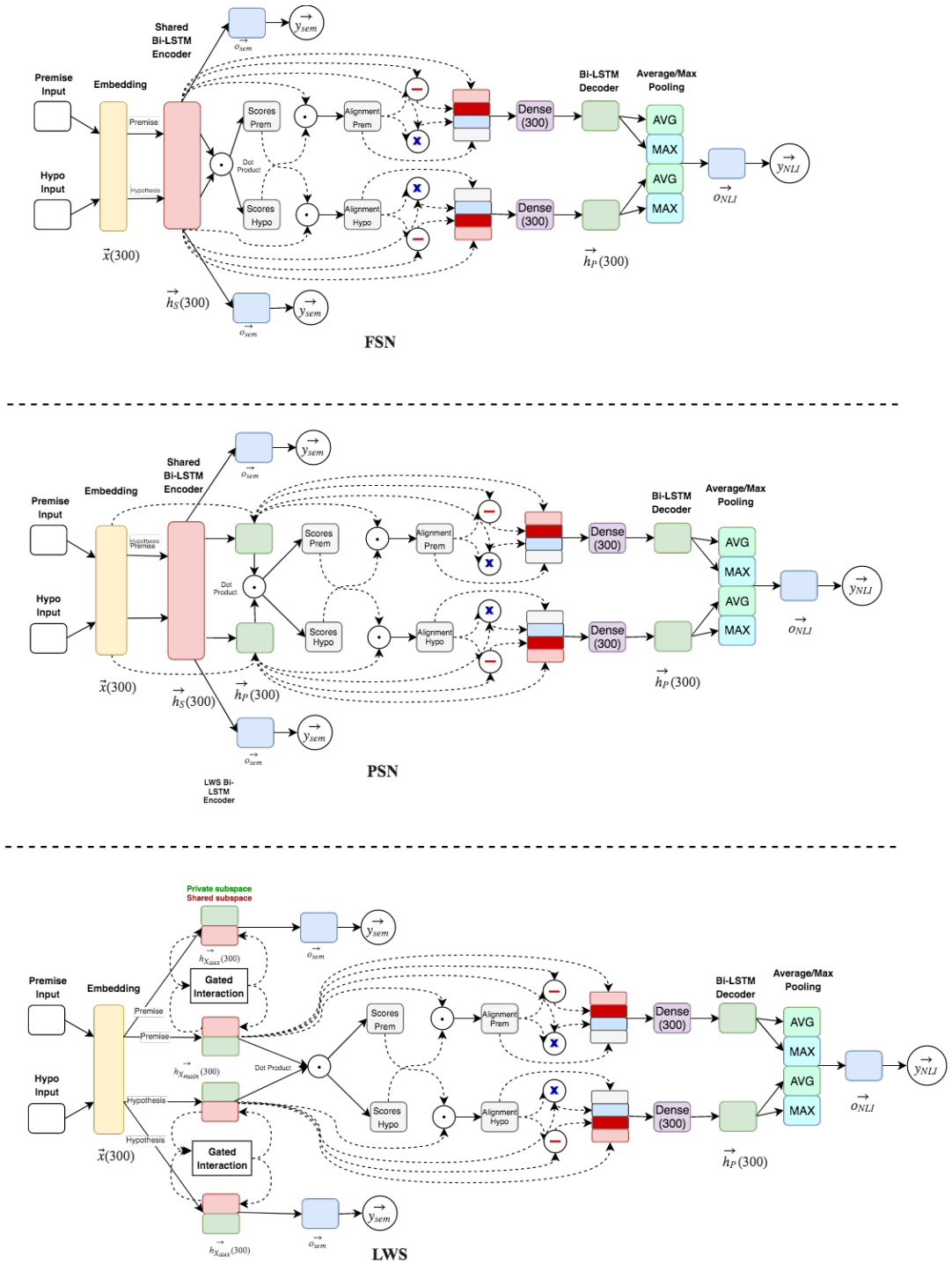



Figure 3.3: The three MTL settings for the NLI system. Layers dimensions are displayed in brackets.

## 3.5 Results

Results for all tasks are shown in Table 3.3. ST indicates the single-task baseline model. FSN, PSN, and LWS are the three MTL settings. Scores are reported as accuracy for the two NLI tasks, and as LAS/UAS $F_1$ score for UD DEP, as is standard.

| Model | SNLI | SICK-E | UD DEP |
|---|---|---|---|
| ST | 87.01 | 81.30 | 80.24 / 84.87 |
| FSN | 84.96 | 56.69 | 81.03 / 85.54 |
| PSN | 87.08 | 77.92 | 80.92 / 85.81 |
| LWS | **87.51** | **84.57** | **81.39 / 86.00** |

Table 3.3: Results for single-task models (ST), fully-shared networks (FSN), partially-shared networks (PSN), and *Learning What to Share* (LWS). All scores are reported as accuracy, except UD DEP for which we report LAS/UAS $F_1$ score.

## 3.6 Analysis

Similar to our findings for UPOS in Chapter 2, the FSN setting leads to an improvement for UD DEP. Indeed, for UD DEP, all of the MTL models outperform the ST model by increasing margins. For the NLI tasks, however, there is a clear degradation in performance.The PSN setting shows mixed results and does not show a clear advantage over FSN for UPOS and UD DEP. This suggests that adding task-specific layers after fully-shared ones does not always enable sufficient task specialization. For the NLI tasks however, PSN is clearly preferable to FSN, especially for the small-sized SICK-E dataset where the FSN model fails to adequately learn.

As a sentence-level task, NLI is functionally dissimilar to semantic tagging. However, it is a task which requires deep understanding of natural language semantics and can therefore conceivably benefit from the signal provided by semantic tagging. Our results demonstrate that it is possible to leverage this signal given a selective sharing setup where negative transfer can be minimized. Indeed, for the NLI tasks, only the LWS setting leads to improvements over the ST models.The improvement is larger for the SICK-E task which has a much smaller training set and therefore stands to learn more from the semantic tagging signal. For all tasks, it can be observed that the LWS models outperform the rest of the models. This is in line with our expectations with the findings from previous work Ruder et al. [2017], Liu et al. [2016] that selective sharing outperforms full network and partial network sharing.

#### SNLI model output

Table 3.4 shows demonstrative examples from the SNLI test set on which the *Learning What to Share* (LWS) model outperforms the single-task (ST) model. The examples cover all possible combinations of entailment classes. Table 1.1 explains the semantic tagset, for reference. It can be observed that the LWS setting correctly predicts the labels in a variety of cases where the ST model does not.

| Premise-hypothesis pairs | ST | LWS/GOLD |
|---|---|---|
| **P**: The$^{\text{DEF}}$ gentleman$^{\text{CON}}$ is$^{\text{NOW}}$ speaking$^{\text{EXS}}$ while$^{\text{SUB}}$ the$^{\text{DEF}}$ others$^{\text{ALT}}$ are$^{\text{NOW}}$ listening$^{\text{EXS}}$<br>**H**: The$^{\text{DEF}}$ man$^{\text{CON}}$ is$^{\text{NOW}}$ being$^{\text{EXS}}$ given$^{\text{EXS}}$ respect$^{\text{CON}}$ | N | E |
| **P**: Men$^{\text{CON}}$ wearing$^{\text{EXG}}$ hats$^{\text{CON}}$ walk$^{\text{EXS}}$ on$^{\text{REL}}$ the$^{\text{DEF}}$ street$^{\text{CON}}$<br>**H**: The$^{\text{DEF}}$ men$^{\text{CON}}$ having$^{\text{EXS}}$ hats$^{\text{CON}}$ on$^{\text{REL}}$ their$^{\text{HAS}}$ head$^{\text{CON}}$ | C | E |
| **P**: Three$^{\text{QUC}}$ men$^{\text{CON}}$ in$^{\text{REL}}$ orange$^{\text{IST}}$ suits$^{\text{CON}}$ are$^{\text{NOW}}$ doing$^{\text{EXG}}$ street$^{\text{CON}}$ repairs$^{\text{CON}}$ at$^{\text{REL}}$ night$^{\text{CON}}$<br>**H**: Three$^{\text{QUC}}$ men$^{\text{CON}}$ in$^{\text{REL}}$ orange$^{\text{IST}}$ suits$^{\text{CON}}$ escaped$^{\text{EPS}}$ from$^{\text{REL}}$ prison$^{\text{CON}}$ | N | C |
| **P**: A$^{\text{DIS}}$ toddler$^{\text{CON}}$ sits$^{\text{ENS}}$ on$^{\text{REL}}$ a$^{\text{DIS}}$ stone$^{\text{CON}}$ wall$^{\text{CON}}$ surrounded$^{\text{EXS}}$ by$^{\text{REL}}$ fallen$^{\text{EXS}}$ leaves$^{\text{CON}}$<br>**H**: An$^{\text{DIS}}$ child$^{\text{CON}}$ is$^{\text{NOW}}$ throwing$^{\text{EXG}}$ stones$^{\text{CON}}$ at$^{\text{REL}}$ a$^{\text{DIS}}$ leaf$^{\text{CON}}$ wall$^{\text{CON}}$ | E | C |
| **P**: An$^{\text{DIS}}$ old$^{\text{IST}}$ shoemaker$^{\text{CON}}$ in$^{\text{REL}}$ his$^{\text{HAS}}$ factory$^{\text{CON}}$<br>**H**: The$^{\text{DEF}}$ shoemaker$^{\text{CON}}$ is$^{\text{NOW}}$ wealthy$^{\text{IST}}$ | C | N |
| **P**: A$^{\text{DIS}}$ kid$^{\text{CON}}$ slides$^{\text{CON}}$ down$^{\text{IST}}$ a$^{\text{DIS}}$ yellow$^{\text{COL}}$ slide$^{\text{CON}}$ into$^{\text{REL}}$ a$^{\text{DIS}}$ swimming$^{\text{CON}}$ pool$^{\text{CON}}$<br>**H**: The$^{\text{DEF}}$ kid$^{\text{CON}}$ is$^{\text{NOW}}$ playing$^{\text{EXS}}$ at$^{\text{REL}}$ the$^{\text{DEF}}$ waterpark$^{\text{CON}}$ | E | N |

Table 3.4: Examples of the entailment problems from SNLI which are incorrectly classified by the ST model but correctly classified by the LWS model. Automatically assigned semantic tags are in superscript.

# Chapter 4

# Multi-lingual Learning as an instance of Multi-task learning

In this chapter we consider the framing of multi-lingual learning as a special case of multi-task learning where every language can be taken a task. Our investigation focuses on the utilization of data from a well-resourced language (English) to lower-resourced languages (German and Dutch). Our aim in this chapter is to answer the following research question:

> **Research Question 4:** *Can multi-lingual approaches which treat multi-lingual learning as a special case of multi-task learning be used to improve semantic tagging accuracy for languages with less or no training data?*

## 4.1 Method

In order to answer the fourth research question, we design a set of experiments to test whether the large semantic tagging dataset we have for English can be used to improve on the semantic tagging models for Dutch and German which are trained on monolingual data only. This is motivated by the fact that only a small amount of data is available for both Dutch and German. We use word embedding mapping methods to facilitate transfer from English.

Our experiment design aims to test the effect of cross-lingual transfer with increasing amounts of in-language monolingual data (i.e. Dutch and German), starting with the zero-shot setting where there is no monolingual data and progressing till the setting where the full monolingual data is used. This is then also compared to the in-language only models, which are trained on the Dutch or German training data only. Below is a list of the settings:

a) **Single language:** In-language monolingual data only (all available data).

b) **Zero-Shot:** No in-language monolingual data.

c) **Quarter:** A quarter of the available monolingual data is used.

d) **Half:** Half of the available monolingual data is used.

e) **Full:** All the available monolingual data is used.

In each of these settings except for the Single Language one, the full training split of the English semantic tagging dataset (75927 sentences) is used.

## 4.2 Data

The dataset we utilize for our semtagging experiments is again derived from the Parallel Meaning Bank (PMB) dataset. For English we use the dataset described in Section 2.1.1. The PMB aslo includes a smaller amount of Dutch and German data. This is also divided into three parts: gold data which is fully manually corrected and silver data which is partially manually corrected. Table 4.1 shows the number of tokens and sentences in each of the three parts for both German and Dutch.

| Language | Part | Gold | Silver |
|---|---|---|---|
| German | Sentences | 1509 | 515 |
| | Tokens | 8520 | 6137 |
| Dutch | Sentences | 692 | 131 |
| | Tokens | 4077 | 1695 |

Table 4.1: The Universal Semantic Tag dataset version 0.1.0's gold and silver parts in number of tokens and sentences for Dutch and German.

Since the amount of gold is rather large when compared to the silver data which is of lower quality, we exclude the silver data from experiments. The

entirety of the gold data is used for is split into training, testing, and development sets. The composition of the train, development, and test sets is shown in table 4.2.

| Language | Split | Test | Train | Development |
|---|---|---|---|---|
| German | Sentences | 493 | 809 | 204 |
| | Tokens | 2849 | 4538 | 1134 |
| Dutch | Sentences | 230 | 372 | 89 |
| | Tokens | 1358 | 2173 | 546 |

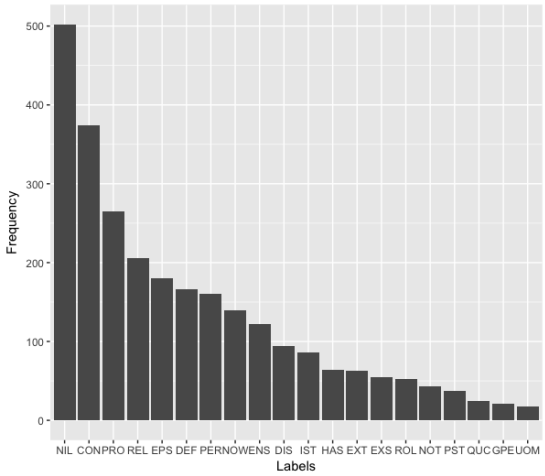
Table 4.2: Test, train, and development splits for both the Dutch and German data.

The dataset is tokenised using the Elephant tokeniser, with seperate models trained on each of the languages. The distributions of the twenty most frequent semtags in the train, development, and testing sets are shown in Figures 4.1a, 4.1b, and 4.1c for German and Figures 4.2a, 4.2b, and 4.2c for Dutch. Unlike in the English data, we observe a consistency across splits. This is due to us not using silver standard data which is only partially manually corrected in this set of experiments, instead employing gold standard data which is fully manually corrected for all three splits.

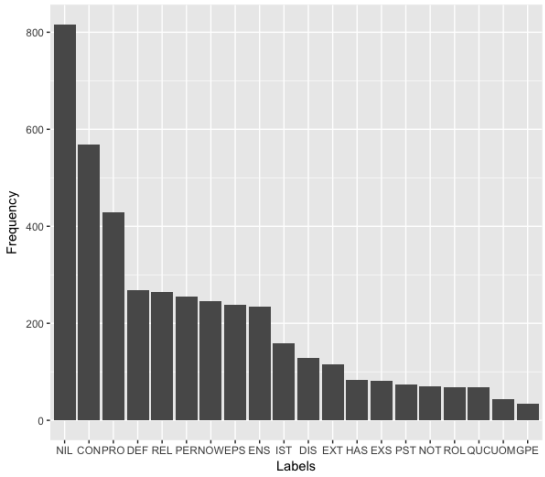
## 4.3 System

As in Section 2.2.5, our tagging model in this set of experiments uses a basic contextual one-layer bi-LSTM (see Section 1.2.5) that takes in word embeddings and produces a sequence of recurrent states which can be viewed as contextualized representations. Our MTL setting is again the fully-shared network setting, but this time we treat each language as a task. To accomplish this, the embedding spaces are unified using a linear mapping and the model then takes in the combined training sets of both languages as input.

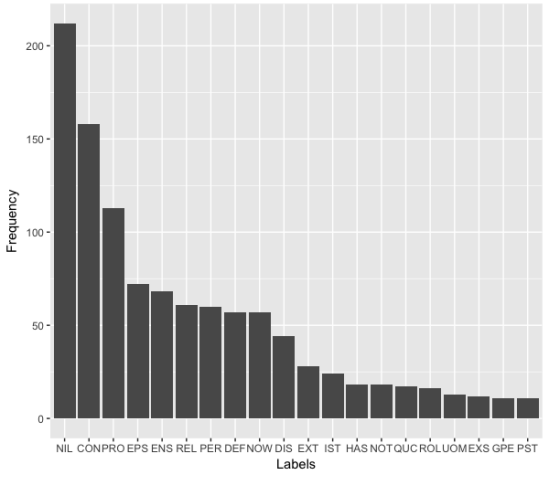
Unlike the tagger in Section 2.2.5, the embeddings layer is a concatenation of both frozen pre-trained word embeddings and randomly initialized task-specific word embeddings. Freezing the pre-trained mutli-lingual word embeddings ensures the models do not forget the cross-lingual relations learned through embeddings mapping. The trainable randomly initialized embeddings ensure the model can also learn task specific word representations, while maintaining the cross-lingual signal. The recurrent $r_n$ state from the bi-LSTM corresponding to each time-step $t_n$ is passed through a dense layer with a softmax activation to predict a token's tag.




(a) Test set distribution

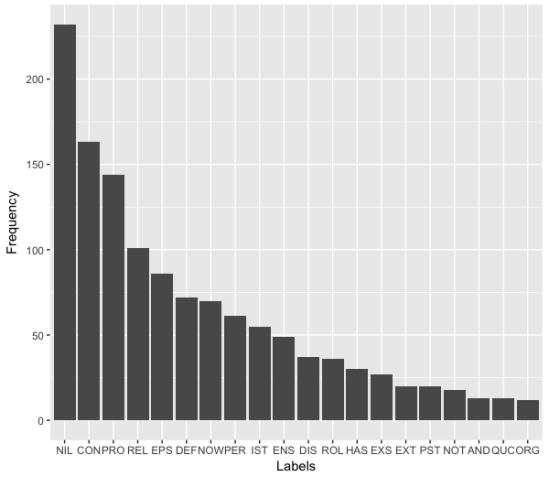



(b) Train set distribution

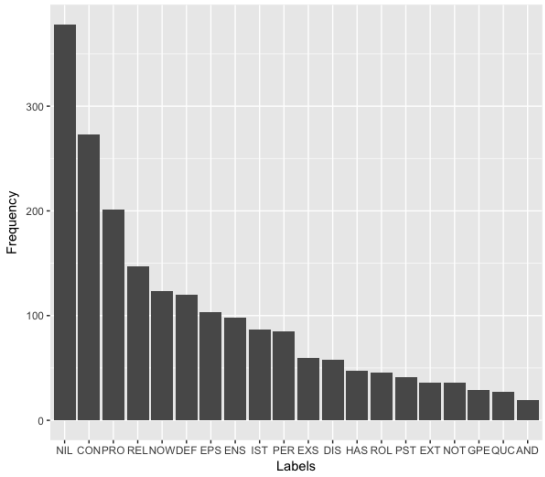



(c) Development set distribution

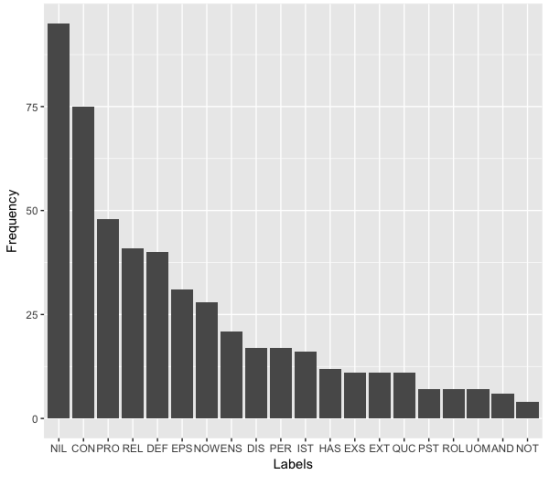
Figure 4.1: The frequency distributions of the twenty most frequent semtags in the test, train, and development sets of the German semantic tagging data.




(a) Test set distribution




(b) Train set distribution




(c) Development set distribution

Figure 4.2: The frequency distributions of the twenty most frequent semtags in the test, train, and development sets of the Dutch semantic tagging data.

### Embedding Mapping

The word embeddings we utilize are FastText embeddings trained on Wikipedia Grave et al. [2018], which are available for 157 languages. We employ the word embedding mapping methods mentioned in 1.5 to map the Dutch and German embedding spaces to the English embedding space. Specifically, we employ the linear mapping method of Artetxe et al. [2016] which makes use of bilingual dictionaries, aiming to learn an optimal linear transformation between to spaces by minimizes the distances between the equivalent entries listed in a bilingual dictionary.

#### Hyperparameters, Optimization, and Initialization

Hyperparameters were tuned with respect to loss on the German and Dutch semantic tagging validation sets. The optimal settings found for both languages were very similar. They are listed below:

- Batch Size: 128
- Epochs: 20
- Optimizer: Adam
- Learning rate: 0.00005
- Embedding initialization: Pre-trained FastText embeddings of dimension 300 (trained on Wikipedia).
- Dropout: with a probability of 0.3 to all bi-LSTM, embedding layers, and non-output dense layer

## 4.4 Results

Results for all tasks are shown in Table 4.3. The `Single language` results are obtained by training and testing on the in-language (i.e. Dutch or German). The `Zero-shot` results are obtained by training on the full English data and testing on either Dutch or German. The results for the `Quarter`, `Half`, and `Full` settings are obtained by training on the full English data and either a quarter, a half, or the full Dutch or German data.

| Setting | German | Dutch |
|---|---|---|
| Single language | 85.41 | 67.52 |
| Zero-shot | 35.01 | 32.13 |
| Quarter | 77.87 | 75.06 |
| Half | 83.64 | 75.82 |
| Full | **88.74** | **81.43** |

Table 4.3: Results for the Single language, Zero-shot, Quarter, Half, and Full settings. All scores are reported as accuracy.

## 4.5 Analysis

The results from our experiments can be seen as an indication of the validity of the embedding alignment method which was utilized to align the vector spaces of the embeddings as a method of enabling cross-lingual transfer. Indeed, The `Zero-shot` models which only see English data at training time achieve an accuracy which is far above random. The results from the `Quarter` setting are the most interesting. With just a quarter (93 sentences) of the in-language training set, the Dutch `Quarter` models outperform the `Single language` Dutch model which is trained on the entire Dutch training set. This has important implications for languages where only a very small amount of training data is available, demonstrating that with less than a hundred sentences and labelled data from a related language, it is possible to achieve a reasonable performance on the task.

Furthermore, it can be seen that the `Full` models which train on both the full English and full in-language training sets outperform the `Single language`. This true even for German, for which the training set is relatively large. This shows that even in cases where a sufficient amount of in-language data is available to train a reasonably accurate model, adding data from a similar language still can help.

# Conclusion

This thesis is an exponent of the recent trend in machine learning in general and representation learning in particular, to design models which simultaneously learn multiple tasks rather than a single task. This has been shown to result in richer representations which are 'generally good' when compared to the representations obtained by learning a single task. In the context of Natural Language Processing and Natural Language Understanding, this approach has recently been applied to an increasingly diverse combination of tasks with reasonable success.

As neural network methods have become a mainstay in the field, a multi-task learning approach in which a network's entire set of parameters is shared between multiple tasks became commonly employed due to ease of implementation. This approach can succeed at times when the tasks being jointly learned are closely related, however, with more distant tasks negative transfer is very likely. Instead, recent work - this thesis included - has explored ways of allowing multi-tasking models to learn what to share and what not to share between a set of tasks. These selective sharing methods offer a general approach to multi-task learning that can be extended to a large number of tasks which can all inform each other even if they are only loosely related. In the larger context of artificial intelligence, this approach presents a promising range of possibilities for moving away from narrow artificial intelligence towards a more general and human-like artificial intelligence.

Our investigation of multi-task learning methods in this thesis was carried out through the lens of a recently proposed task of semantic tagging. In this task each word is assigned a tag representing its semantic class. Our choice of task is motivated by the intuition that capturing lexical semantic distinctions can be informative for a variety of Natural Language Processing tasks. Our investigation is broken down into four related research questions, each addressed in one of the chapters. Below are the conclusions which can be drawn from each of the four chapters [1].

## Multi-task Learning for Sequence Labelling Tasks

In chapter 2 we began by addressing the following research question:

> **Research Question 1:** *Which sequence labelling tasks, if any, can help with semantic tagging in a multi-task learning setting?*

We explore the utility of three different tasks as auxiliaries for semantic tagging: Universal Dependencies Part-Of-Speech Tagging, Combinatory Categorial

[1] Chapter I is an introduction

Grammar supertagging, and Universal Dependencies dependency relation tagging. We choose these tasks based on previous usage as auxiliaries for semantic tasks in the literature. Moreover, we introduce a novel highway network layer to replace a residual network layer in a state-of-the-art deep residual semantic tagger, giving the model more flexibility in controlling residual connections. We find that out of the three tasks only Universal Dependencies Part-Of-Speech Tagging in combination with highway networks results in an improvement over the single-task model which does not use an auxiliary. This agrees with some of the previous findings regarding the characteristics of the label distributions of tasks which perform well as auxiliaries for semantic tasks: a low kurtosis and a compact label distribution. Overall, our results point towards the limitations of the fully-shared network approach to multi-task learning.

We then turn to the converse of the first question aiming to answer the following question:

> **Research Question 2:** *Can semantic tagging be informative for other sequence labelling tasks? If so, how and under which multi-task settings?*

Observing the limitations of fully-shared networks in the previous set of experiments, we introduce two more flexible mutli-task learning settings: partially-shared networks and a setting we term *Learning what to share*. To answer **Research Question 2** we employ semantic tagging as an auxiliary task for Universal Dependencies Part-Of-Speech Tagging, using a bidirectional LSTM tagging model which is then adapted to each of the mutli-task learning settings.

Our two main findings are: a) all multi-task settings show an improvement over the single-task model, meaning that the semantic tagging task is informative for Universal Dependencies Part-Of-Speech Tagging in all cases and b) the *Learning what to share* setting outperforms all other settings by a considerable margin, showing that even for tasks which are close enough for fully-shared networks to work, a more nuanced approach to parameter sharing still leads to better performance.

In this chapter and the next, we offer a generalizable framework for the evaluation of the utility of an auxiliary task under different multi-task learning settings. This is important because the majority of previous work on the evaluation of how useful certain auxiliary tasks are only uses the FSN setting [Plank, 2016b, Søgaard and Goldberg, 2016] which we show is inferior to other settings. Other more recent work which presents selective sharing architectures focuses primarily on the those architectures themselves, without paying too much attention to the auxiliary tasks themselves [Liu et al., 2016, Meyerson and Miikkulainen, 2017, Ruder et al., 2017]. We offer a unification of the two strands of work by evaluating the utility of semantic tagging in different settings.

## Multi-task Learning for Sentence-level and Structured Prediction Tasks

We then turn our attention to sentence-level and structured prediction tasks aiming to answer the following the following question:

**Research Question 3:** *Can semantic tagging be informative for higher-level semantic tasks such as natural language inference or structured prediction tasks such as dependency parsing in a multi-task learning or transfer learning setting?*

We take the tasks of Universal Dependency parsing and Natural Language Inference as main tasks and employ semantic tagging as an auxiliary. Our use of a sequence labelling auxiliary task with sentence-level (Natural Language Inference) and structured prediction (Universal Dependency parsing) tasks is novel. As far as we are aware, this combination has only been attempted by Hashimoto et al. [2016] which take a hierarchical approach and do not employ selective sharing mechanisms.

Our findings show that for Universal Dependency parsing all multi-task learning settings were beneficial and that the more flexible settings lead to a larger improvement. For Natural Language Inference the picture is more complex. As Natural Language Inference only has labels on the sentence level rather than on the word-level, it is structurally unlike semantic tagging. However, it is also a task which requires deep understanding of natural language semantics and can therefore conceivably benefit from the signal provided by semantic tagging. Our results show that it is indeed possible to leverage this signal given a selective sharing setup where negative transfer can be minimized. Indeed, only the most flexible of out settings leads to improvements over the single-task models for this task.

## Multi-lingual Learning as an instance of Multi-task learning

Finally we turn our attention to multi-lingual learning, where multi-lingual learning is framed as a special case of multi-task learning with each language treated as a task. Our goal is to answer the following question:

**Research Question 4:** *Can multi-lingual approaches which treat multi-lingual learning as a special case of multi-task learning be used to improve semantic tagging accuracy for languages with less or no training data?*

In order to answer **Research Question 4**, we design a set of experiments to test whether the semantic tagging dataset which is available for English can be used to improve the semantic tagging models for Dutch and German. Thus is motivated by the fact that a much larger amount of data English than for German and Dutch in the parallel meaning bank.

We use word embedding mapping methods to facilitate cross-lingual transfer by aligning the embedding spaces across languages. Our overall findings indicate that this is a viable approach to transferring information between between languages: a) our zero-shot models which only see English training data show an above chance performance and b) with only a very small amount of in-language training data, the multi-lingual models can outperform the models which are trained on the full monolingual data.

## Future Work

There are many directions in which this work can be extended:

- Exploring the broad space of other tasks which can benefit from semantic tagging as an auxiliary.
- Analyzing the results achieved and representations obtained by multi-task models in order attain a more solid understanding of why multi-task learning helps.
- Further examining the conditions (mutli-task learning settings, task combinations, etc.) under which multi-task and transfer learning can be successful.
- Developing more flexible methods of multi-task learning. In this thesis we showed that selective sharing methods hold an advantage over less flexible methods. Further exploration of this direction of research promises to yield multi-task learning methods which could allow for effective joint learning of a large number of diverse tasks.
- As more effective multi-task learning methods are developed, the potential for large-scale multi-lingual learning using the methods demonstrated in this thesis will become more and more possible. This will allow for low-resource languages to benefit from the advances made in Natural Language Processing which are currently to a large extent only available for high-resource languages.

# Bibliography


Lasha Abzianidze and Johan Bos. Towards universal semantic tagging. In *Proceedings of the 12th International Conference on Computational Semantics (IWCS 2017) – Short Papers*, pages 1–6, Montpellier, France, 2017a.

Lasha Abzianidze and Johan Bos. Towards universal semantic tagging. *arXiv preprint arXiv:1709.10381*, 2017b.

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik Van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. The parallel meaning bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. *arXiv preprint arXiv:1702.03964*, 2017.

Héctor Martínez Alonso and Barbara Plank. When is multitask learning effective? semantic sequence prediction under varying data conditions. *arXiv preprint arXiv:1612.02251*, 2016.

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A Smith. Many languages, one parser. *arXiv preprint arXiv:1602.01595*, 2016.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, 2016.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. *arXiv preprint arXiv:1805.06297*, 2018.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P14-1023`.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. *arXiv preprint arXiv:1801.07772*, 2018.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3 (Feb):1137–1155, 2003.

Joachim Bingel and Anders Søgaard. Identifying beneficial task relations for multi-task learning in deep neural networks. *arXiv preprint arXiv:1702.08303*, 2017.

Johannes Bjerva. One model to rule them all: Multitask and multilingual modelling for lexical analysis. *arXiv preprint arXiv:1711.01100*, 2017.

Johannes Bjerva and Isabelle Augenstein. Tracking typological traits of uralic languages in distributed language representations. *CoRR*, abs/1711.05468, 2017. URL `http://arxiv.org/abs/1711.05468`.

Johannes Bjerva, Barbara Plank, and Johan Bos. Semantic tagging with deep residual networks. *arXiv preprint arXiv:1609.07053*, 2016.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642. Association for Computational Linguistics, 2015a. doi: 10.18653/v1/D15-1075. URL `http://www.aclweb.org/anthology/D15-1075`.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. Generating sentences from a continuous space. *CoRR*, abs/1511.06349, 2015b. URL `http://arxiv.org/abs/1511.06349`.

Thorsten Brants. Tnt: a statistical part-of-speech tagger. In *Proceedings of the sixth conference on Applied natural language processing*, pages 224–231. Association for Computational Linguistics, 2000.

Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

R Caruna. Multitask learning: A knowledge-based source of inductive bias. In *Machine Learning: Proceedings of the Tenth International Conference*, pages 41–48, 1993.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750, 2014.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1657–1668, 2017.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Yoeng-Jin Chu. On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400, 1965.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Stephen Clark and James R Curran. The importance of supertagging for wide-coverage ccg parsing. In *Proceedings of the 20th international conference on Computational Linguistics*, page 282. Association for Computational Linguistics, 2004.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *CoRR*, abs/1103.0398, 2011. URL `http://arxiv.org/abs/1103.0398`.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *CoRR*, abs/1705.02364, 2017a. URL `http://arxiv.org/abs/1705.02364`.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017b.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017c.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1107–1116, 2017d.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*, 2014.

Timothy Dozat and Christopher D Manning. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*, 2016.

Timothy Dozat, Peng Qi, and Christopher D Manning. Stanford's graph-based neural dependency parser at the conll 2017 shared task. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, 2017.

Jack Edmonds. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240, 1967.

Jeffrey L. Elman. Finding structure in time. *COGNITIVE SCIENCE*, 14(2): 179–211, 1990.

Katrin Erk. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653, 2012. URL `http://dblp.uni-trier.de/db/journals/llc/llc6.html#Erk12`.

Kilian Evang, Valerio Basile, Grzegorz Chrupała, and Johan Bos. Elephant: Sequence labeling for word and sentence segmentation. In *EMNLP 2013*, 2013.

J. R. Firth. A synopsis of linguistic theory 1930-55. 1952-59:1–32, 1957.

Jonas Gehring, Michael Auli, David Grangier, and Yann N Dauphin. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344*, 2016.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

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

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

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Jeremy Howard and Sebastian Ruder. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146, 2018. URL `http://arxiv.org/abs/1801.06146`.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 873–882, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2390524.2390645`.

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

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *AAAI*, pages 2741–2749, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. Improving sentence compression by learning to predict gaze. *CoRR*, abs/1604.03357, 2016. URL `http://arxiv.org/abs/1604.03357`.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10): 1995, 1995.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*, 2016.

Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, Yu Cheng, Tara Javidi, and Rogerio Feris. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5334–5343, 2017.

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

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. A sick cure for the evaluation of compositional distributional semantic models. In *LREC*, pages 216–223, 2014.

Elliot Meyerson and Risto Miikkulainen. Beyond shared hierarchies: Deep multitask learning through soft layer ordering. *arXiv preprint arXiv:1711.00108*, 2017.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013a. URL `http://arxiv.org/abs/1301.3781`.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013b.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013c.

Marvin Minsky, Seymour A Papert, and Léon Bottou. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3994–4003, 2016.

Ndapa Nakashole and Raphael Flauger. Characterizing departures from linearity in word translation. *arXiv preprint arXiv:1806.04508*, 2018.

Joakim Nivre, Lars Ahrenberg Zeljko Agic, et al. Universal dependencies 2.0. lindat/clarin digital library at the institute of formal and applied linguistics, charles university, prague, 2017.

Robert Östling. Morphological reinflection with convolutional neural networks. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 23–26, 2016.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

Barbara Plank. Keystroke dynamics as signal for shallow syntactic parsing. *CoRR*, abs/1610.03321, 2016a. URL `http://arxiv.org/abs/1610.03321`.

Barbara Plank. What to do about non-standard (or non-canonical) language in nlp. *arXiv preprint arXiv:1608.07836*, 2016b.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *CoRR*, abs/1604.05529, 2016. URL `http://arxiv.org/abs/1604.05529`.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Sluice networks: Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142*, 2017.

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

Anders Søgaard and Yoav Goldberg. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 231–235, 2016.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. On the limitations of unsupervised bilingual dictionary induction. *arXiv preprint arXiv:1805.03620*, 2018.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015. URL `http://arxiv.org/abs/1505.00387`.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. *arXiv preprint arXiv:1804.00079*, 2018.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4499–4509, 2017.

Kristina Toutanova and Christopher D Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 63–70. Association for Computational Linguistics, 2000.

Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *J. Artif. Int. Res.*, 37(1):141–188, January 2010. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=1861751.1861756`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017a. URL `http://arxiv.org/abs/1706.03762`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017b.

Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavior science. *Unpublished Doctoral Dissertation, Harvard University*, 1974.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, 2015.

Yuan Zhang, David Gaddy, Regina Barzilay, and Tommi Jaakkola. Ten pairs to tag-multilingual pos tagging via coarse mapping between embeddings. Association for Computational Linguistics, 2016.

# List of Figures

# List of Tables