Department of Computer Science
Series of Publications A
Report A-2023-10

# Neural Transfer Learning for Truly Low-Resource Natural Language Processing

Eliel Soisalon-Soininen

*Doctoral dissertation, to be presented for public examination with the permission of the Faculty of Science of the University of Helsinki in Auditorium PIII at Porthania, Yliopistonkatu 3, on the 6th of July 2023 at 12 o'clock.*

University of Helsinki
Finland

**Supervisor**
Hannu Toivonen, University of Helsinki, Finland
Mark Granroth-Wilding, Silo AI

**Pre-examiners**
Christina Lioma, University of Copenhagen, Denmark
Jaakko Peltonen, Tampere University, Finland

**Opponent**
Liviu Dinu, University of Bucharest, Romania

**Custos**
Hannu Toivonen, University of Helsinki, Finland

**Contact information**

Department of Computer Science
P.O. Box 68 (Pietari Kalmin katu 5)
FI-00014 University of Helsinki
Finland

Email address: info@cs.helsinki.fi
URL: http://cs.helsinki.fi/
Telephone: +358 2941 911

Copyright © 2023 Eliel Soisalon-Soininen
ISSN 1238-8645 (print)
ISSN 2814-4031 (online)
ISBN 978-951-51-9343-8 (paperback)
ISBN 978-951-51-9342-1 (PDF)
Helsinki 2023
Unigrafia

# Neural Transfer Learning for Truly Low-Resource Natural Language Processing

Eliel Soisalon-Soininen

Department of Computer Science
P.O. Box 68, FI-00014 University of Helsinki, Finland
eliel.soisalon-soininen@helsinki.fi

PhD Thesis, Series of Publications A, Report A-2023-10
Helsinki, June 2023, 121 pages
ISSN 1238-8645 (print)
ISSN 2814-4031 (online)
ISBN 978-951-51-9343-8 (paperback)
ISBN 978-951-51-9342-1 (PDF)

## Abstract

The vast majority of the world's languages are low-resource, lacking the data resources required in advanced natural language processing (NLP) based on data-intensive deep learning. Furthermore, annotated training data can be insufficient in some domains even within resource-rich languages. Low-resource NLP is crucial for both the inclusion of language communities in the NLP sphere and the extension of applications over a wider range of domains. The objective of this thesis is to contribute to this long-term goal especially with regard to *truly low-resource* languages and domains.

We address truly low-resource NLP in the context of two tasks. First, we consider the low-level task of *cognate identification*, since cognates are useful for the cross-lingual transfer of many lower-level tasks into new languages. Second, we examine the high-level task of *document planning*, a fundamental task in data-to-text natural language generation (NLG), where many domains are low-resource. Thus, domain-independent document planning supports the transfer of NLG across domains. Following recent encouraging results, we propose neural network models to these tasks, using *transfer learning* methods in three low-resource scenarios.

We divide our high-level objective into three research tasks characterised by different resource conditions. In our first research task, we address cognate identification in endangered Sami languages of the Uralic family, given scarce labelled training data. We propose a Siamese convolutional neural network (S-CNN) and a support vector machine (SVM), which we pre-train on unrelated Indo-European data, lacking high-resource close relatives. We find that S-CNN performs best at direct transfer to Sami, and adapts fast when fine-tuned on a small amount of Sami data. In our second research task, we address a scenario with only unlabelled data to adapt S-CNN from Indo-European to Uralic data. We propose both discriminative adversarial networks and pre-trained symbol embeddings, finding that adversarial adaptation outperforms an unadapted model, while symbol embeddings are beneficial when languages have disparate orthographies.

In our third research task, we address document planning in data-to-text generation of news, in a domain with no annotated training data whatsoever. We propose distant supervision, automatically constructing labelled data from a news corpus, and train a neural model for sentence ordering, a task related to document planning. We examine Siamese, positional, and pointer networks, and find that a variant of S-CNN results in generation with higher human-perceived quality than heuristic baselines.

The contributions of this thesis include addressing novel low-resource scenarios considering two NLP tasks, at which the potential of deep learning has not been fully explored. We propose novel approaches to these tasks using neural models in combination with transfer learning, and our experiments indicate their performance in comparison with baselines. Finally, although we acknowledge that rule-based methods and heuristics might still be superior to deep learning in truly low-resource scenarios, our approaches are more language- and domain-independent, supporting a wider coverage of NLP across languages and domains.

**Computing Reviews (2012) Categories and Subject Descriptors:**

Computing Methodologies → Artificial Intelligence → Natural Language Processing
Computing Methodologies → Machine Learning → Machine Learning Approaches→ Neural Networks
Computing Methodologies → Artificial Intelligence → Natural Language Processing → Natural Language Generation

**General Terms:**
Algorithms, Design, Experimentation, Languages

**Additional Key Words and Phrases:**
natural language processing, low-resource scenarios, neural networks, transfer learning, cognate identification, document planning

# Acknowledgements

What a long journey it has been, and never could I anticipate what would await me during these last few years. First and foremost, I would like to convey my thanks to my supervisors, Professor Hannu Toivonen and Dr Mark Granroth-Wilding, for their guidance throughout my doctoral studies. I extend my thanks to my colleagues both in the Discovery Group and the doctoral programme (DoCS) for their company and peer support.

I would like to express my gratitude to the Academy of Finland and the European Union for funding my work through the *Digital Language Typology* (grant no. 12933481) and *EMBEDDIA* (grant no. 825153) research projects, inspiring the topics of this thesis. I also thank the Helsinki Institute of Information Technology (HIIT) and the Finnish Center for Artificial Intelligence (FCAI) for providing me with the aid I needed to finally reach the finish line. In addition, I thank Professor Peter Widmayer and his research group for hosting my visit at ETH Zũrich.

I am grateful to my father Eljas for his guidance, as well as to my mother Sari and my brother Ilmari for their encouragement. I also thank the rest of my family and friends for their support and prayers. Finally, my deepest thanks go to my wife Maria for her endless love and patience, and to our little Efraim for bringing so much joy into our life.

Pernå, June 2023
Eliel Soisalon-Soininen

# Contents

# Chapter 1

# Introduction

The vast majority of the world's over 7,000 languages are currently being left behind from the increasingly impressive developments of *natural language processing* (NLP) and *computational linguistics* [63]. While advanced high-level applications are built for English and a handful of other languages using data-intensive deep learning, the same cannot be done for most languages, as their data resources are insufficient. Furthermore, similar data scarcity can apply to certain domains and tasks even within a resource-rich language [55]. NLP for such *low-resource* scenarios has been named as one of the most important open problems in the field [122], and is crucial to support the inclusion of language communities in the NLP sphere, and to increase its coverage across different domains and tasks.

The availability of language-, domain-, or task-specific data resources varies greatly within the low-resource category. While some languages have potential resources left unexploited despite a relatively large speaker population [49], others are endangered with tiny (if any) resources [63], lacking even the most basic NLP tools. Meanwhile, extremely low-resource domains and tasks, with no suitable training data whatsoever, might arise especially when data requirements are particularly great and annotation is expensive. Apart from low-resource languages, this can be the case for certain high-level applications even in high-resource languages [22]. We refer to this kind of extreme scenarios as *truly low-resource.*

## 1.1 Motivation

In this thesis, we consider two different NLP tasks in truly low-resource scenarios. First, we address the low-level, linguistic task of identifying etymologically related words, *cognates*, between truly low-resource languages.

Cognate information can support the implementation and transfer of low-level NLP tasks into new languages, which is of utmost importance in order to eventually build high-level applications for a larger set of languages. Cognate information has been found useful for low-level tasks including the processing of non-standard orthographies [35], morphological analysis [47, 51, 96], part-of-speech tagging [125], as well as word-level translation [53, 100]. Cognates are also useful for language-learning applications [10], and of course, they are essential for historical linguistics [89].

Second, we focus on the higher-level task of content selection and ordering, known as *document planning*, for natural language generation (NLG). Document planning is a fundamental task in NLG [121], which in turn is a core part of high-level NLP applications. Since this task has been relatively little addressed in NLG research despite its difficulty [37], many domains even within high-resource languages are truly low-resource with little annotated data [57, 137]. The coverage of NLG across a wider range of domains, and eventually languages, is encouraged by document planning solutions that are more robust and domain-independent [48].

The potential of deep learning and neural networks has not been fully explored in the context of our tasks, especially in truly low-resource scenarios. Indeed, the lower-resource an NLP scenario is, the more difficult it is to train data-intensive models. Therefore, statistical [e.g 95, 111], and especially rule-based [e.g. 46, 149] approaches might result in better outcomes in some cases. Nevertheless, along with some encouraging results for both cognate identification [67] and document planning [114], in this thesis we are interested in applying deep-learning models to these tasks. In contrast to rule-based methods, models learning patterns from data do not require language- or domain-specific expertise, but are instead more scalable to other languages and domains. Although statistical methods have similar benefits, neural networks tend to outperform them after a relatively low threshold of data requirements is met [95, 111], or sometimes by reducing neural network complexity [34]. At the very least, even if deep learning models are not viable in some extreme scenarios, they can still be valuable in combination with more traditional methods [149, 152].

In the context of cognate identification and document planning, we consider three truly low-resource scenarios with different resource conditions. First, we address cognate identification across endangered Uralic languages with a tiny amount of labelled data for the task. Second, while addressing the same task, we consider languages with only unlabelled data. Third, we address document planning for data-to-text generation of statistical news, a domain with no suitable training data for the task. As these scenarios involve little or no language- or domain-specific training data, we cannot train neural networks with such data directly. Instead, our approach is to leverage the limited available data together with related, auxiliary data.

*Transfer learning* refers to the *pre-training* of models on auxiliary data from a *source* language or domain, and transferring them to a *target* language or domain, using a range of techniques depending on the scenario. This principle has been inspired by the human way of learning — if one already knows English, it is hardly necessary (or even possible), to start language learning completely from scratch when starting to learn French. Recent research indicates that transfer learning achieves promising results in truly low-resource scenarios [33, 87, 110]. Furthermore, neural networks making use of transfer learning generally outperform supervised models that do not [123].

## 1.2 Research Tasks

The purpose of this thesis is to expand the field of NLP to previously unaddressed truly low-resource scenarios. We consider three kinds of novel scenarios, each characterised by different resource conditions with regard to the involved tasks, languages, and domains. We address two NLP tasks, cognate identification and document planning, that have received relatively little research attention overall in the past, and no previous work to our knowledge does this for the same set of languages and domains. Furthermore, although neural networks are the dominant tool of choice in NLP research, we find that for these tasks, their potential has not been fully explored. At the same time, recent research implies a high potential of transfer learning in combination with neural networks. Hence, our objective is to develop neural-network solutions for our tasks, using transfer learning to adapt them to our low-resource scenarios. We divide this objective into the following research tasks (RTs).

**Research Task I:**

*Given scarce language-specific labelled data, identify cognates in truly low-resource Sami languages.*

In our first low-resource scenario, we address the task of cognate identification between three truly low-resource Uralic languages, namely South, North, and Skolt Sami. These represent our *target* language set, and our data for these languages consists of small vocabularies and a small set of verified cognates between them. As this is alone insufficient for supervised training, our aim is to extend this with auxiliary data from *source* languages, in our case from the Indo-European language family.

**Research Task II:**

*Given only unlabelled data, adapt a pre-trained cognate identification model to truly low-resource Uralic languages.*

In our second low-resource scenario, we continue with cognate identification between truly low-resource Uralic languages. However, we consider a case where our language-specific data is completely unlabelled. Our aim is to pre-train models on source languages, and to adapt them to our target languages using unlabelled data only.

**Research Task III:**

*Given only auxiliary data, develop a method for document planning in news generation.*

In our third low-resource scenario, we are concerned with the task of document planning within a data-to-text news generation pipeline. Document planning consists of content selection and document structuring, that is, determining what information is presented in a generated output and in which order. Although we consider the high-resource English language, direct supervised training for the task is not possible, due to a lack of annotated training data with aligned data–text samples. Our aim is to construct training data from auxiliary data, an unannotated news text corpus from a similar domain.

## 1.3 Contributions

Having presented the background of this thesis in Chapters 2–3, we devote each of Chapters 4–6 to one research task (RT). In these three chapters, we make the following contributions.

In Chapter 4, we address our first research task **RT-I**. To accomplish this, we first propose two models, a support vector machine (SVM) and a Siamese convolutional neural network (S-CNN). These learn a string metric from data, which is suitable for predicting cognacy between words. In contrast to most previous work that assumes the availability of high-resource relatives, we consider transfer across highly unrelated language families. We *pre-train* the models on Indo-European etymological data, and test them on Uralic languages from the Sami group. In comparison with a string-metric baseline, we examine the models' ability to generalise to Sami, and find that S-CNN outperforms SVM and the baseline, Finally, we exploit transfer learning by *fine-tuning* S-CNN on a small amount of labelled Sami data, to quantify how well it can make use of such data to adapt. We find its performance improves fast already with little fine-tuning.

Chapter 4 is based on a published conference paper [132], and a workshop abstract describing the setup and initial results [133]. Both of these papers were co-authored with supervisor Mark Granroth-Wilding, who provided feedback and suggestions in both the planning and writing of the papers. In the early stages of this work, the author of this thesis also collaborated with Mika Hämäläinen, presenting together at a conference an abstract discussing cognate identification in the context of Sami languages [134].

In Chapter 5, we are concerned with our second research task **RT-II**. To accomplish this, we begin with the same neural model, S-CNN, as we propose in Chapter 4. As then, we pre-train the model on Indo-European etymological data, but now we assume that no labelled data for Uralic languages is available whatsoever. Therefore, we propose two unsupervised methods: one of *domain adaptation*, and another of *cross-lingual adaptation*. The former is based on latent feature learning: using *adversarial networks* to make S-CNN's latent representations of source and target data more similar to each other. The latter uses pre-trained *cross-lingual symbol embeddings*, creating a common representation space for source and target languages. To evaluate the efficacy of our proposed methods, we compare the performance of adapted and unadapted models at identifying cognates between different truly low-resource Uralic languages. We find that adversarial adaptation outperforms an unadapted model, and that pre-trained symbol embeddings provide benefits in certain scenarios.

In Chapter 6, we address our third research task **RT-III**. To accomplish this, we propose *distant supervision*: we use auxiliary data, a news text corpus, to create training data for the task of *sentence ordering*, our *source task* related to our *target task*, document planning. We examine three kinds of neural models for this source task: Siamese, positional, and pointer networks. Due to its performance and relatively low computational complexity, we choose to incorporate a Siamese convolutional network (a variant of S-CNN mentioned above), in an existing news generation pipeline. We conduct a human evaluation of generation outputs, and find that our neural approach produces outputs rated higher by evaluators than those of our baselines.

The work in Chapter 6 has been a collaboration with Leo Leppänen, the developer of the news generation system [84] to which we apply our neural approach. He contributed to the chapter by implementing the baseline document planners in Section 6.4 and assisting with the design of the human evaluation.

## 1.4 Thesis Outline

This thesis consists of seven chapters. From this introduction, we proceed to present the background of this thesis in Chapters 2–3. First, in Chapter 2, we lay out the theoretical background, focusing on neural networks, transfer learning, and low-resource NLP. Then, in Chapter 3, we describe the specific NLP tasks examined in this thesis, cognate identification and document planning, and provide brief literature reviews concerning these tasks.

Chapters 4–6 present our contributions, addressing our research tasks RT-I, RT-II, and RT-III. In each chapter, we present the specific problem setting, our proposed methods and experiments with them, summarising the findings at the end of each chapter. Finally, in Chapter 7, we conclude the thesis with a discussion of our findings.

# Chapter 2

# Background

In this chapter, we present the theoretical background of this thesis. First, we explain the fundamentals of neural networks, and give a brief overview of models most relevant to our work. We then proceed to describe a theoretical framework of transfer learning, considering different learning scenarios. In the last section, we review approaches to low-resource NLP, particularly with regard to truly low-resource scenarios.

## 2.1 Neural Networks in NLP

*Neural networks* are a family of machine learning models based on collections of connected computational units, *neurons*, inspired by the neural networks found in biological brains. Through stacked layers of connected neurons, neural networks facilitate the learning of complex representations from simple ones, which is referred to as *deep learning*. During the last decades, neural networks have become the most popular tool of choice in NLP. This development has followed from the general success achieved with neural network models, which has been a result of increasing computational power and dataset sizes, allowing for the training of larger models with more learning capacity. In the following subsections, we describe several kinds of neural network models used especially in NLP and this thesis.

### 2.1.1 Basics

The fundamental example of a neural network is the *feed-forward network*, or *multi-layer perceptron* (MLP), and it is the basis of most neural networks. It can be regarded as a function approximation $\hat{f}$ of some underlying function $f$, mapping an input $\mathbf{x}$ to an output $\hat{\mathbf{y}} = \hat{f}(\mathbf{x}; \boldsymbol{\theta})$. Given some *training* data

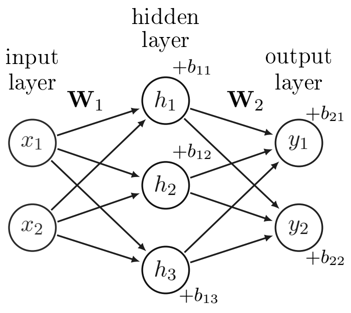

Figure 2.1: An MLP network with one hidden layer. There are two neurons in input and output layers, and three neurons in the hidden layer. The weight matrices $\mathbf{W}_1, \mathbf{W}_2$ assign weights to each connection between layers.

as examples of input–output pairs, the goal is to learn the parameters $\boldsymbol{\theta}$ that approximate $f$ as well as possible.

The MLP maps inputs to outputs using a sequence of *layers*: an *input* layer, one or more *hidden* layers, and an *output* layer, each consisting of one or more neurons. Figure 2.1 depicts a simple MLP network with one hidden layer. Information is fed forward in the network through intermediate computations at each neuron in the hidden layer(s) and the output layer. In this example, starting from an input vector $\mathbf{x} = [x_1, x_2]$, the hidden layer vector $\mathbf{h} = [h_1, h_2, h_3]$ is computed such that

$$\mathbf{h} = g_1(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$$

where $\mathbf{W}_1 \in \mathbb{R}^{3\times 2}$ is a *weight* matrix between the input and hidden layers, $\mathbf{b}_1$ is the layer's *bias* vector, and $g_1$ its *activation function*. The weight matrix contains the weights assigned to each connection between the input and hidden layers' neurons, determining the importance of different input features, while the bias vector $\mathbf{b}_1 = [b_{11}, b_{12}, b_{13}]$ contains the constant parameters of each hidden-layer neuron. The activation function $g_1$ applies a non-linearity to each neuron-wise sum $wh + b$. The final output of this MLP is then

$$\hat{f}(\mathbf{x}; \boldsymbol{\theta}) = \hat{\mathbf{y}} = g_2(\mathbf{W}_2\mathbf{h} + \mathbf{b}_2)$$

where $\mathbf{W}_2 \in \mathbb{R}^{2\times 3}$ is the weight matrix between the hidden and output layers, $\mathbf{b}_2$ is the output layer's bias vector, and $g_2$ is the output layer's activation function.

In general, an MLP with $n$ layers is a chain of $n$ functions $\hat{f}(\mathbf{x}; \boldsymbol{\theta}) = f^n(f^{n-1}(\dots f^{(1)}(\mathbf{x})))$ where the function of layer $i$ is $f^{(i)} = g_i(\mathbf{W}_i\mathbf{h}_{i-1} + \mathbf{b}_i)$. The model's parameters $\boldsymbol{\theta}$ consist of the weights $\mathbf{W}_i$ and biases $\mathbf{b}_i$, which are learned by minimising a *loss* $L(\hat{\mathbf{y}}, \mathbf{y})$, a function of the error between the model's predictions and desired outputs. The objective is to minimise this loss on unseen test data using a separate training dataset, usually assumed to follow the same underlying data distribution. This minimisation is based on computing the loss function's gradient $\nabla_{\boldsymbol{\theta}} L$ using the back-propagation algorithm, and updating the model's parameters towards a direction that is opposite to the one indicated by the gradient, a technique known as gradient descent [43].

### Activation Functions

A non-linear activation function is essential for enabling the learning of complex functions by the network, since without one, it reduces to linear regression. In this thesis, we use the following four activations in our models: the rectified linear unit (ReLU), sigmoid ($\sigma$), hyperbolic tangent (tanh), and softmax, defined as

$$\begin{aligned}
\mathrm{ReLU}(x) &= \max\{0, x\} \\
\sigma(x) &= \frac{1}{1+e^{-x}} \in [0,1] \\
\tanh(x) &= 2\sigma(2x) - 1 \in [-1,1] \\
\mathrm{softmax}(\mathbf{x}) &= \frac{e^{x_i}}{\sum_j e^{x_j}} \in [0,1]^{|\mathbf{x}|}
\end{aligned}$$

of which we use ReLU in hidden layers as it is easy to optimise, and the others in the output layers of our models. In this thesis, we use the sigmoid for binary classification in both cognate identification and sentence ordering, as its output can be regarded as the probability that an input belongs to the positive class, which can be converted to a discrete prediction using a certain probability threshold. In Chapter 6, we use the hyperbolic tangent in our recurrent models, as is common due to its slower tendency to zero in longer sequences. We use the softmax in the same chapter for multi-class classification in the positional and pointer networks, as the function essentially provides a probability distribution over several classes, where the predicted class is considered the one with the highest probability.

### Word Embeddings

In NLP, an important use of basic MLP networks is in training *word embeddings*, dense representations of words in a common vector space, where words with similar meanings are close to each other [97, 107]. That is, a word (string) $w$ in some vocabulary $V$ is mapped to a vector $\mathbf{x}$, providing a numerical input to a neural network. One of the most popular methods to compute word embeddings is *word2vec*, based on shallow MLP networks trained on a corpus of text to predict either the context of a given word, or a word given its context. In addition to words, embeddings can also be computed of other language units, such as characters [16, 44, 139] and sub-words [15]. More recently, more complex Transformer-based language models have been developed for the computation of high-queality word embeddings [32].

### 2.1.2 Recurrent Networks

The *recurrent* neural network (RNN) differs from the MLP in that it has been designed for processing sequential inputs $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. In this thesis, we propose recurrent long short-term memory networks and their bidirectional variant for sentence ordering in Chapter 6.

The main principle of the RNN is that at each time step $t$, in addition to processing a new input $x_t$, it receives information from previous time steps and passes it on to the next one. Such connections from one time step to the next can be assigned between different layers of the network, most commonly between its hidden layers. Such an RNN is illustrated in Figure 2.2. That is, the network's *hidden state* $\mathbf{h}$ is updated at each time step $t$ based on the previous hidden state $\mathbf{h}_{t-1}$ and the input $x_t$. The hidden states and outputs are then computed as

$$\mathbf{h}_t = \tanh(\mathbf{V}\mathbf{h}_{t-1} + \mathbf{W}_1\mathbf{x}_t + \mathbf{b}_1)$$

$$\mathbf{y} = g(\mathbf{W}_2\mathbf{h}_t + \mathbf{b}_2)$$

where $g$ is output activation, $\mathbf{W}_1, \mathbf{W}_2, \mathbf{V}$ are the weights, and $\mathbf{b}_1, \mathbf{b}_2$ are the biases. The difference from MLP is the additional weight matrix $\mathbf{V}$ between hidden states. In a *bidirectional* RNN [126], the hidden state consists of two layers, one for each direction of the sequence, which has been found to preserve information better while doubling the hidden state's number of parameters.

#### Language Models

A *language model* generates sequences of language units, usually at the level of words or characters, given some context as input. Language models constitute a fundamental component of many higher-level NLP solutions [18, 32, 108]. A common neural language model is an RNN which,

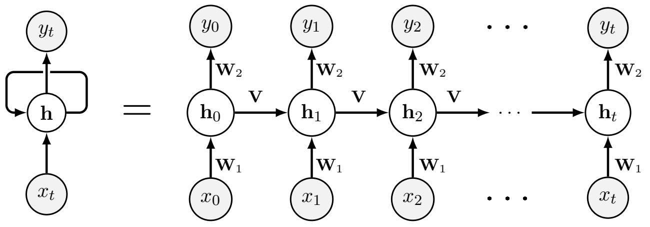

Figure 2.2: A recurrent neural network with connections between hidden states.

given a sequence of word embeddings $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_n)$, aims to learn the distribution

$$P(\mathbf{X}) \approx \prod_{t=1}^{n} P(\mathbf{x}_t | \mathbf{x}_{t-1}, \dots, \mathbf{x}_0),$$

that is, the probability of the sequence as a product of the conditional probabilities of previous words. The same principle applies to Transformer-based language models, which we use in Chapter 6.

### LSTM Networks

*Long short-term memory* (LSTM) networks [56] have been designed to capture long-term dependencies more effectively than the basic RNN. To do this, the hidden state $\mathbf{h}_t$ is complemented with a *cell state* $\mathbf{C}_t$ and several *gates*, which control the flow of information in the network. These gates are a forget gate $f_t$, an input gate $i_t$, and an output gate $o_t$, which are all associated with their own parameters with regard to the previous hidden state $\mathbf{h}_{t-1}$ and input $\mathbf{x}_t$. These are computed similarly to $\mathbf{h}_t$ above, but using a sigmoid activation to produce values between 0 and 1. The cell state $\mathbf{C}_t$ is then updated as

$$\mathbf{C}_t = f_t \mathbf{C}_{t-1} + i_t \times \tanh(\mathbf{V}\mathbf{h}_{t-1} + \mathbf{W}_1 \mathbf{x}_t + \mathbf{b}_1)$$

and the new hidden state as $\mathbf{h}_t = o_t \times \tanh(\mathbf{C}_t)$, controlled by the output gate. As RNNs, LSTMs can also be bidirectional or arranged into a sequence-to-sequence architecture, and have been used for training language models [108].

### Sequence-to-Sequence Networks

Most notably used in neural machine translation and other generation tasks, a *sequence-to-sequence* architecture consists of two recurrent networks: an

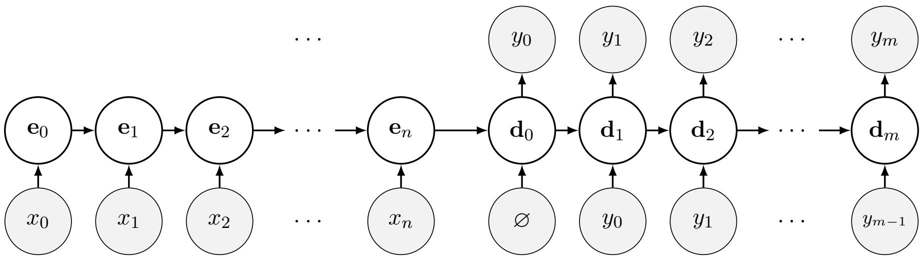

Figure 2.3: A sequence-to-sequence network with RNN encoder and decoder.

*encoder* and a *decoder*. In this thesis, we use such an architecture in the pointer network in Chapter 6 to encode shuffled sentences and decode them in the correct order. Sequence-to-sequence networks have been most notably applied to machine translation.

While the encoder works as the basic RNN described above, its purpose is to encode the input sequence $\mathbf{x} = (x_1, \ldots, x_n)$ into its last hidden state $\mathbf{e}_n$, as illustrated by Figure 2.3. This input sequence representation is then used to initialise the decoder's hidden state, that is $\mathbf{d}_0 = \mathbf{e}_n$. Given initially a start token (e.g. a random vector), the decoder predicts the first output element $y_0$. Using the previous output as the next input, the decoder predicts the output sequence $\mathbf{y} = (y_0, \ldots, y_m)$. As opposed to using each output as the next input, a globally more probable output can be obtained using algorithms such as beam search.

Since sequence-to-sequence RNN networks also struggle to capture long-term dependencies, the *attention* technique has been proposed to alleviate the bottleneck problem of more recent inputs being weighted more. The attention mechanism is based on computing a specific representation of encoder hidden states, to which the softmax activation is applied. This results in element-wise attention weights for the inputs, which can be used to compute a new hidden state for the decoder. Different variants of this mechanism exist, including additive [7] and multiplicative [92] attention, of which we use the former in the pointer network in Chapter 6.

### 2.1.3 Convolutional Networks

The *convolutional* neural network (CNN) is specialised for grid-like data such as images, but they have also been found to perform well at certain NLP tasks such as sentence classification [129] and character-based language modelling [70, 158]. Likewise, we find a convolutional network to perform well at our tasks in this thesis.

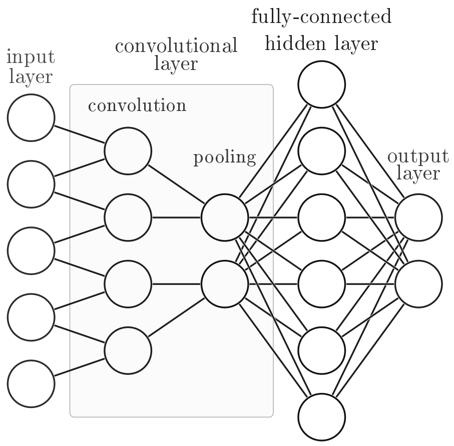

Figure 2.4: A convolutional neural network with one convolutional layer, consisting of both one convolution (cross-correlation) and a pooling layer, followed by one fully-connected hidden layer. Both convolution and pooling widths are equal to two nodes.

A convolutional layer usually consists of three operations: convolution, activation, and pooling. In contrast to the MLP with fully-connected layers, in the convolution a kernel with weights $\mathbf{W} \in \mathbb{R}^{i \times j}$ is cross-correlated with equal-sized portions of the input $\mathbf{X} \in \mathbb{R}^{n \times m}$, in the two-dimensional case. This is illustrated in Figure 2.4 with a one-dimensional CNN, where each node in the convolutional layer is connected only to two nodes of the input layer. An activation function, usually ReLU, is applied to the resulting output, followed by a pooling operation. For example, max-pooling takes the maximum value within a certain window, equal to two nodes in Figure 2.4. This convolutional layer consisting of both a convolution and a pooling operation is usually followed by a fully-connected hidden layer, and finally the output layer.

The benefit of convolutional layers is that the kernel requires fewer parameters and is equivariant to the input. That is, since the convolutional kernel is moved through the grid-like input in a sliding window, the weights are less affected by features' exact positions in the input. In addition to many computer vision applications, this is useful even for our tasks in this thesis. In cognate identification, we are more interested in whether certain substrings are present in a word than in their position within the word (see Section 3.1). In addition, in Chapter 6, we observe that a convolutional network outperforms a recurrent one at sentence classification.

### 2.1.4 Multiple-Input Networks

While we have so far addressed architectures where one input is processed at a time, certain tasks require the comparison of several inputs simultaneosly. The most common form of such *multiple-input* networks are *Siamese* networks considering two inputs at a time. Such consist of two identical networks processing their respective inputs simultaneously, producing a hidden-layer representation of each. These representations can then be compared using a similarity metric, e.g. cosine distance, or merged into one representation and fed forward to a following layer as a single input.

Multiple-input architectures can be constructed from different types of networks, including Siamese variants of feed-forward and recurrent networks for NLP [24, 130] as well as convolutional networks for learning image similarity [26]. In this thesis, we propose both convolutional and recurrent Siamese networks for our tasks, both to learn cognacy between words (Chapters 4–5) and to order sentences (Chapter 6).

### 2.1.5 Transformers

More recently, the *Transformer* [144] architecture has become prevalent in NLP, outperforming recurrent neural networks in language modelling and a wide range of downstream tasks. In particular, language models trained on massive datasets are based on Transformers, and have been found to perform well when fine-tuned for specific tasks [18]. The Transformer is based on giving up the sequential architecture and using instead *self-attention* and positional encodings for tokens. As a sequence-to-sequence model, it consists of an encoder and a decoder. While both are required for generative tasks such as language modelling, the encoder can be used alone to compute embeddings of language tokens, usually words [32].

The encoder's architecture is illustrated in Figure 2.5. Given an initial input sequence of tokens, the embedding layer turns each token into a fixed-size embedding. In order to keep track of the inputs' order in the sequence, each input embedding is summed with a positional encoding indicating a token's relative position using sine and cosine functions. The result is then fed to a stack of encoder layers, where each encoder layer consists of two sub-layers: multi-head self-attention and a feed-forward hidden layer, each one followed by layer-normalisation [6]. The multi-head attention layer uses several heads of self-attention, that is, dot-product attention between each token and all other tokens in the same sequence, allowing the model to efficiently compute context-dependent representations. The Transformer's encoder is the basis for doing this in models such as BERT [32]. In this

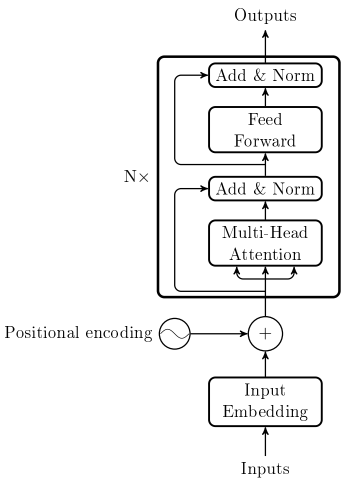

Figure 2.5: The encoder of the Transformer architecture [144].

thesis, we use its variant, Sentence-BERT [120], to obtain embeddings of sentences and tokens for our document planning task in Chapter 6.

When used as a sequence-to-sequence architecture, such as in language modelling, the outputs of the Transformer's encoder are fed to a stack of decoders layers. Similarly to the encoder layers, each decoder layer consists of self-attention and feed-forward hidden layers followed by layer-normalisation. However, between these two sub-layers is an additional hidden layer computing attention between encoder and decoder representations. The decoder layer stack produces a representation corresponding to the predicted next token given the inputs, which is then fed to a fully-connected layer followed by a softmax layer, finally indicating the actual token predicted. This is then taken as input by the next decoding step.

## 2.2 Transfer Learning

In traditional supervised learning, a model is first trained on labelled data for a certain task, and it is then expected to perform well at this task given some unseen test data. This paradigm is based on the assumption that both training and test datasets have the same feature spaces and underlying data distributions, and that there is only one task. Given another test dataset with a different feature space or data distribution, or a different task, a new model has to be trained on a new training dataset. This is illustrated in Figure 2.6, where models $\mathcal{M}_A$ and $\mathcal{M}_B$ are trained and tested on their respective datasets $D_A$ and $D_B$, for their respective tasks $\mathcal{T}_A$ and $\mathcal{T}_B$.

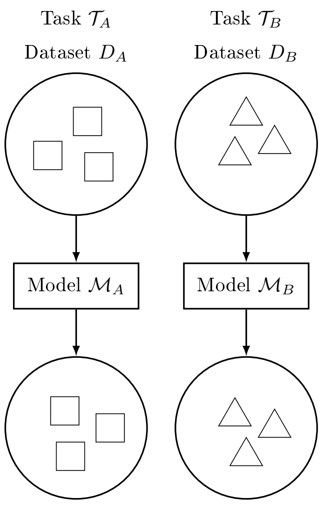

Figure 2.6: Supervised learning.

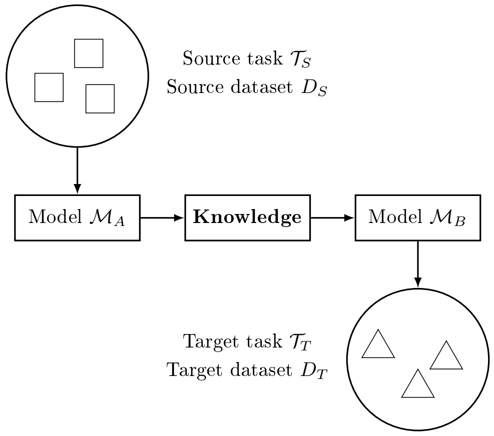

Figure 2.7: Transfer learning.

In real-world applications, it is common that some of these assumptions regarding datasets and tasks do not hold, and that a new model cannot be trained easily. For example, labelled data in dataset $D_B$ might be insufficient for training model $\mathcal{M}_B$ for task $\mathcal{T}_B$. However, given that dataset $D_A$ is to some extent related to $D_B$, model $\mathcal{M}_A$ can be trained on $D_A$ for task $\mathcal{T}_A$, the knowledge gained transferred to model $\mathcal{M}_B$, which can be applied to dataset $D_B$ and task $\mathcal{T}_B$. This is the paradigm of *transfer learning*, illustrated in Figure 2.7, where a model is trained on *source data* for a *source task*, and applied to *target data* and a *target task*. In the context of neural networks, the transferred knowledge is typically in the form of learned representations.

### 2.2.1 Definition

Transfer learning addresses machine learning scenarios where there is a mismatch between source and target data or source and target tasks. Pan and Yang [103] propose a theoretical framework for transfer learning, which was further developed by Ruder [123]. They define a *domain* $\mathcal{D}$ as a pair

$$\mathcal{D} = \{\mathcal{X}, P(X)\}$$

where $\mathcal{X}$ is a feature space and $P(X)$ is the marginal probability distribution over data $X \in \mathcal{X}$. In the context of NLP, $\mathcal{X}$ is usually a language, represented as a set of words or characters, of which $X$ is a sample distributed according to $P(X)$. For example, $P(X)$ could represent a text genre. In NLP, the term *domain* usually refers only to $P(X)$. In this thesis we adopt this convention, and refer to $\mathcal{D}$ instead as a *learning domain.*

Given a learning domain $\mathcal{D}$, a *learning task* $\mathcal{T}$ is defined as

$$\mathcal{T} = \{\mathcal{Y}, P(Y), P(Y|X)\}$$

where $\mathcal{Y}$ is a label space, $P(Y)$ is a prior distribution over the labels, and $P(Y|X)$ is a conditional distribution over the labels given data $X$. Typically, $P(Y|X)$ is learned from a training dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. For example, in binary classification tasks, the label space is $\mathcal{Y} = \{0, 1\}$.

A transfer learning scenario is characterised by a source learning domain $\mathcal{D}_S$ and a corresponding task $\mathcal{T}_S$, as well as a *target* learning domain $\mathcal{D}_T$ and a corresponding task $\mathcal{T}_T$. The objective is then to learn the target conditional distribution $P_T(Y_T|X_T)$ using information learned from $\mathcal{D}_S$ and $\mathcal{T}_S$, when $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$. Otherwise, if $\mathcal{D}_S = \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$, the problem reduces to supervised learning and no transfer learning is needed.

### 2.2.2 Conditions

From the definitions above it follows that there are in total five components where $\mathcal{D}_S$ and $\mathcal{D}_T$ or $\mathcal{T}_S$ and $\mathcal{T}_T$ can differ, giving rise to various transfer learning scenarios with their respective solution approaches. Corresponding to these components, the following conditions can occur either alone or together with others in one learning scenario.

(1) $\mathcal{X}_S \neq \mathcal{X}_T$: Data samples come from different feature spaces. In NLP, this is typical when training and test datasets are in different languages. We face this condition in Chapters 4–5, where we transfer cognate identification models from one set of languages to another.

(2) $P_S(X_S) \neq P_T(X_T)$: The marginal distributions of source and target data are different. This condition is known as *domain shift*, which we face throughout this thesis: in Chapters 4–5, our source and target data are from different languages, and since there is no one-to-one mapping between the feature spaces, the distributions are different.

(3) $\mathcal{Y}_S \neq \mathcal{Y}_T$: The source and target tasks' label spaces are different. We face this condition in Chapter 6, where our source task is sentence ordering and our target task is document planning. Meanwhile, in Chapters 4–5, source and target label spaces are both binary.

(4) $P_S(Y_S) \neq P_T(Y_T)$: The prior distributions of source and target labels are different. This is the case in both of the tasks we address in this thesis. In Chapters 4–5, while label spaces are the same, they are differently distributed in source and target datasets. In Chapter 6, this condition follows from condition 3.

(5) $P_S(Y_S|X_S) \neq P_T(Y_T|X_T)$: The conditional distributions of source and target labels are different given the data. That is, there is a class imbalance between source and target datasets. A common condition, which we also observe in each of Chapters 4–6.

### 2.2.3 Taxonomy

Pan and Yang [103] propose a taxonomy of transfer learning approaches addressing the above conditions. Its adaptation to NLP by Ruder [123] is shown in Figure 2.8. In this taxonomy, the two main categories of transfer learning are *transductive* and *inductive* learning. In inductive transfer learning, the domains are the same, $\mathcal{D}_S = \mathcal{D}_T$, while tasks are different, $\mathcal{T}_S \neq \mathcal{T}_T$, at least in some of their components. Transductive transfer learning addresses the opposite case, where $\mathcal{D}_S \neq \mathcal{D}_T$, but $\mathcal{T}_S = \mathcal{T}_T$. In practice, real-life learning scenarios are often a combination of these two categories.

Transductive transfer learning is further divided into two categories, depending on which inequality between source and target domains is present. *Domain adaptation* addresses domain shift, when $P(X_S) \neq P(X_T)$, while *cross-lingual adaptation* addresses scenarios with different source and target languages and disparate feature spaces, $\mathcal{X}_S \neq \mathcal{X}_T$. In this thesis, we deal with transductive transfer learning for cognate identification in Chapters 4–5. As our source and target languages are different, and there is no one-to-one mapping between source and target samples, it follows that the distributions are different as well [123]. In transductive transfer learning, it is assumed that labelled target data is either not available or only in small amounts. This is the case in our low-resource scenarios addressed in Research Tasks RT-I and RT-II (see Section 1.2).

Inductive transfer learning is likewise divided into two categories: *sequential* and *multi-task* learning. This distinction depends on whether source and target tasks are learned sequentially or simultaneously. In Chapter 6, we apply sequential transfer learning to neural document planning. In inductive transfer learning, it is usually assumed that labelled target data is available. Although this is not our case, we create labelled data through distant supervision.

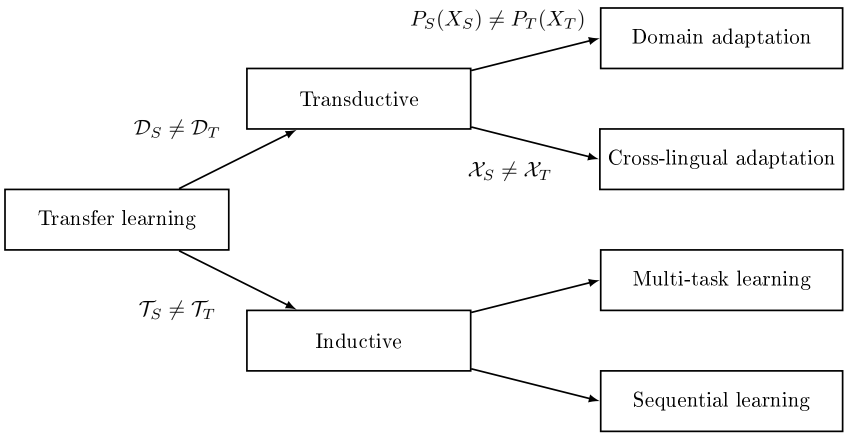

Figure 2.8: Taxonomy of transfer learning approaches, adapted to NLP by Ruder [123] from the original of Pan and Yang [103].

### 2.2.4 Domain Adaptation

In domain adaptation (DA), the aim is to learn a predictive function $f$, usually the target conditional distribution $P_T(Y_T|X_T)$, from a source learning domain $\mathcal{D}_S$ under the domain shift condition $P_S(X_S) \neq P_T(X_T)$. The source and target tasks are the same however, $\mathcal{T}_S = \mathcal{T}_T$. In NLP, example scenarios of domain shift include cases where source and target data have different genres or topics. Although feature spaces are typically the same, $\mathcal{X}_S = \mathcal{X}_T$, they can also be different, a scenario addressed by *heterogeneous* domain adaptation [159]. DA methods can be supervised, unsupervised, or semi-supervised [109], of which we use the first two alternatives in this thesis. Regardless of the degree of supervision, DA methods in NLP can be grouped into two main categories: data-centric and model-centric [119].

*Data-centric* methods include pseudo-labelling, data selection and weighting, as well as pre-training. In pseudo-labelling, a trained classifier is used to predict labels for further training in a semi-supervised setting, while data selection and weighting aims to select or weight data so that performance on target data improves. In this thesis, we use pre-training and fine-tuning for cognate identification in Chapter 4.

*Model-centric* methods can be either feature- or loss-centric. Feature-centric methods include feature augmentation and generalisation, while methods such as domain-adversarial neural networks and loss reweighting belong to the latter category. Feature augmentation represents an earlier supervised approach, where both domain-specific and domain-general features are created [31]. Feature generalisation uses autoencoders to learn lower-dimensional latent representations that are robust across domains and reduce domain shift [40]. Similarly, domain-adversarial networks aim to do the same by simultaneously training a predictor while maximising the confusion of a discriminator trying to distinguish between source and target features [3, 36, 127]. This approach has been found to work well in computer vision [19, 106] but also in cross-lingual NLP [146]. We review the use of this method in low-resource NLP later in Section 2.3.3. In loss reweighting, distribution similarity measures, such as maximum mean discrepancy [45], are used to define loss functions that guide training towards learning domain-general features.

### 2.2.5 Cross-Lingual Adaptation

In cross-lingual adaptation, source and target feature spaces are different, that is $\mathcal{X}_S \neq \mathcal{X}_T$, which occurs in NLP particularly when source and target languages are different. In many real-world transfer learning scenarios, both feature spaces and data distributions are different to some extent. However, in cross-lingual adaptation it is generally assumed there is a correspondence between source and target features, that is, the representations of language units [123]. The objective of cross-lingual adaptation is thus to learn a mapping $f : \mathcal{X}_S \to \mathcal{X}_T$, which allows the approximation of target data distribution as $P_T(X_T) \approx P_S(f(X_S))$.

There are several approaches to cross-lingual adaptation, including multilingual language models [28, 90], and aligned or combined monolingual representations [79]. Pre-trained language models have been found effective especially when pre-trained on large amounts of data in high-resource languages and applied to low-resource ones, even without further adaptation, depending on language relatedness [58, 59, 88]. In such cases, fine-tuning with target-language data brings fast improvements [54, 81].

Alignment of monolingual representations, *embeddings*, results in a common feature space for source and target languages. This can be done at different levels of language units, most commonly words [e.g 1, 58, 88], but also characters [16, 44, 140], sub-words [15, 143], sentences [4], and even languages [58]. Alignments of such embeddings can be learned from multilingual datasets where language-specific data is either parallel or comparable. In the case of words, one prominent way to align monolingual embeddings is by learning a mapping between them [124]. For example, supervised learning from bilingual dictionaries [1] but also unsupervised learning with adversarial training [78] have been applied to this task. Moreover, cognate information has been found to enhance such methods [53].

### 2.2.6 Sequential Learning

In Ruder's [123] taxonomy, sequential transfer learning is specifically concerned with scenarios where $\mathcal{T}_S \neq \mathcal{T}_T$, the source and target tasks differ. However, we note that sequential learning is not only applicable when source and target tasks are different, but can also be considered a method of supervised domain adaptation [119] or cross-lingual adaptation [55], depending on the learning scenario. Sequential learning consists of two stages: *pre-training* and *adaptation*.

There are three approaches to pre-training a model on source data: direct supervision, distant supervision, and unsupervised learning. The most straightforward approach is direct supervised learning using labelled source data, the availability of which varies according to scenario. In distant supervision, labelled training data is automatically obtained from some auxiliary data exploiting heuristics and external information, introduced by Mintz et al. [98]. Finally, unsupervised pre-training requires only unlabelled source data, which makes it the most scalable approach. For example, using pre-trained embeddings for a target task can be considered a form of sequential learning [123].

The two main approaches to adapting a pre-trained model are feature extraction and fine-tuning. In feature extraction, representations learned in pre-training (e.g. embeddings) are directly used as input to the target task, while fine-tuning continues to train the representations using labelled target data. In fine-tuning, it is common to update only a subset of a neural network's layers. For example, fine-tuning large language models for specific tasks has resulted in superior performance across a wide range of NLP tasks [18, 32].

## 2.3 Low-Resource NLP

In this thesis, we apply neural networks and transfer learning to *low-resource* scenarios of NLP. Low-resource NLP is concerned with overcoming resource scarcity utilising a spectrum of methods depending on specific resource conditions. These methods have two main categories [55]. First, there are methods that reduce the need for labelled target data, including pre-training and latent feature learning. Second, there are those that generate additional labelled target data, such as data augmentation, distant supervision, and cross-lingual projection. We give a brief overview of methods relevant to this thesis: fine-tuning a pre-trained model (Chapter 4), domain-adversarial networks (Section 5.2), cross-lingual character-level representations (Section 5.3), and distant supervision (Chapter 6). We consider these in the context of extreme resource conditions, which we refer to as *truly* low-resource scenarios.

### 2.3.1 Resource Availability

In the context of NLP, *resources* refer to the amount and quality of labelled and unlabelled training data as well as NLP tools in a language or domain, which modern applications depend on. The more resources there exist in a certain language, the more complex and higher-level applications can be built. For example in English, high-quality applications exist for difficult high-level tasks such as question answering, while low-resource languages might only have tools for basic tasks such as tokenisation and morphological analysis [55].

Although the term low-resource is widespread in the field, there is no exact definition of the amount of resources that makes a language or domain low-resource. In fact, there is great variance in data resources depending on language, domain, and task, which does not necessarily correlate with the number of language speakers [63]. For example, languages such as Quechuan with 60 million speakers have few resources, while Estonian with one million speakers has relatively large resources. Likewise, even within languages classified as endangered, some have significantly larger resources than others [49]. Ultimately, resource requirements depend on the specific scenario, implying that the concept of low-resource is task-, language-, and domain-specific [55]. That is, even within a high-resource language, resources might be scarce for some domains or tasks.

Hedderich et al. [55] categorise low-resource scenarios according to the availability of three kinds of data:

(1) *Task-specific labels.* Availability of labelled data in target language or domain determines whether direct supervised learning is feasible in a given scenario.

(2) *Unlabelled language- or domain-specific data.* Availability of unlabelled data in target language or domain determines the feasibility of unsupervised training of, for example, language representations.

(3) *Auxiliary data.* Availability of auxiliary data determines the applicability of transfer learning methods using the auxiliary data as their *source* data (see Section 2.2).

In this thesis, we address scenarios that are truly low-resource. That is, they are extreme cases along at least one of the above dimensions of resource availability. In Chapters 4–5, we examine the task of cognate identification between Uralic languages, including definitely endangered North Sami (∼25,000 speakers) and North Karelian (∼20,000 speakers) as well as severely endangered South Sami (∼600 speakers), Skolt Sami (∼300 speakers), and Ingrian (<100 speakers) [99]. We consider also the higher-resource Finnish and Estonian, since they are truly low-resource with regard to our task. Having access to distant auxiliary data in another language family, the Indo-European, we address scenarios with a tiny amount of task-specific labelled target data (Chapter 4) and only unlabelled language-specific data (Chapter 5). In Chapter 6, we focus on the task of document planning for news generation. Although we do this in the high-resource English language, we face a scenario without any task-specific labelled data, but only auxiliary data, which we leverage for our target task using distant supervision. Such scenarios of low-resource domains are common in NLG tasks [57].

### 2.3.2 Pre-Training and Fine-Tuning

Pre-training and fine-tuning is a straightforward and effective approach to supervised adaptation across both languages and domains. In this thesis, we use supervised pre-training and fine-tuning for cognate identification in Sami languages in Chapter 4. This is a case of sequential transfer learning, where a small amount of labelled target data is available, although not sufficiently for direct supervision. Since labelled data for the same task is available in relatively large amounts in unrelated languages of another language family, we use this data for supervised pre-training.

Pre-training and direct transfer alone can already be effective when source languages are high-resource and related to target languages [58, 88]. While direct transfer across language families is challenging [59], fine-tuning has been shown to bring about significant benefits already with small amounts of labelled target-language data at both lower-level and higher-level tasks [54, 81]. Thus, we will investigate the amount of labelled target data required for effective fine-tuning in Chapter 4.

### 2.3.3 Domain-Adversarial Networks

*Domain-adversarial neural networks* (DANs) are a method of latent feature learning, for unsupervised adaptation across domains and languages. DANs have gained more research interest in NLP recently, following the success of similar methods in computer vision. The aim of latent feature learning is to make source and target data more similar to each other by projecting them to a common, lower-dimensional representation space. DANs do this by simultaneously learning a predictor for a given task and maximising the confusion of a discriminator trying to distinguish between source and target representations, using a gradient reversal layer [36].

In NLP, both cross-domain and cross-lingual transfer is possible using domain adversaries. For example, they have been used for domain adaptation in duplicate question detection [65, 127] and social media post classification [3]. In terms of cross-lingual adaptation (or heterogeneous domain adaptation), Chen et al. [23] propose a method in the context of sentiment classification, training a language discriminator adversarially with a feature extractor. Likewise, Kim at al. [68] use a similar method for cross-lingual transfer of POS tagging.

The benefit of domain-adversarial networks is their scalability and generality, as they require only unlabelled target data. Therefore, we propose discriminative adversarial networks in Chapter 5 for cognate identification, following an adaptation method developed for image domains [142].

### 2.3.4 Cross-Lingual Character Representations

As discussed in Section 2.2.5, en effective approach to cross-lingual adaptation is the use of pre-trained language representations, most commonly in the form of word embeddings [55]. However, as we address the low-level task of cognate identification in this thesis, we are more interested in character-level representations. Thus, in Chapter 5, we propose to enhance our neural cognate identification model with symbol embeddings pre-trained using the Xsym model [44].

The quality of word embeddings depends on the language they have been trained on, working inadequately on morphologically-rich languages such as Finnish and Turkish [143]. Thus, instead of considering single words as atomic units, sub-word embeddings [15] and neural character-level language models [70] have been proposed for such cases. Character-level representations have also been found useful for text classification [158] and machine translation [82, 100]. In addition to being useful for different tasks, character representations encode phonological information about languages, being valuable for linguistics [16, 44].

### 2.3.5 Distant Supervision

As a method of sequential transfer learning, *distant supervision* refers to the use of heuristics and external sources of information to construct labelled source data for pre-training, from some unlabelled auxiliary data. In Chapter 6, we do this for the target task of document planning for news generation. From auxiliary data, namely a news corpus, we construct a training dataset for the task of sentence ordering, our source task. (We refer to the unannotated data as *auxiliary* data, and to the constructed labelled training data as *source* data.)

Many forms of distant supervision have been used in NLP, and it is popular in certain tasks with structured knowledge sources, such as information extraction [55]. The heuristics for automatic annotation depend on the use case, ranging from string matching [153] to complicated pipelines [101]. In addition to annotating unlabelled data within the same language and domain, distant supervision has also been used for cross-lingual transfer in part-of-speech tagging, exploiting a combination of annotation projections, lexicons, and representations [110]. Similarly, distant supervision has been found effective for language modelling in truly low-resource languages [54]. Distant supervision has proven useful even for higher-level tasks such as discourse parsing [60], encouraging its use in our document planning task.

# Chapter 3

# Tasks

Low-resource NLP is concerned with various scenarios and tasks, each setting requiring a tailored approach depending on resource availability, as discussed in Section 2.3. In this chapter, we present the specific NLP tasks we address in this thesis: cognate identification, and document planning for news generation. For each task, we provide a brief background and literature review.

## 3.1 Cognate Identification

*Cognate identification* refers to the task of identifying etymologically related words across languages, and it is a core task in historical linguistics [89]. In addition, it is a supportive task for the cross-lingual transfer of many low-level NLP tasks, such as processing non-standard orthographies [35], morphological analysis [47, 51, 96], part-of-speech tagging [125], as well as word-level translation [53, 100]. Cognate information is also useful for language learning [11].

Historical linguistics is the study of language change over time. One of its most important areas is the *comparative method*, which aims to establish historical relatedness between languages by comparing them, especially their words. It is used to group languages into *families*, to infer *phylogenetic trees* reflecting family relationships, and to reconstruct ancient languages, among other historical linguistic tasks. A map over the world's language families is shown in Figure 3.1, together with a phylogenetic tree illustrating how languages within a family descend from a common root language. Although some languages' historical family relationships are well-established, this is not the case for many truly low-resource languages with few speakers and scarce resources. Since language comparison is manually laborious,

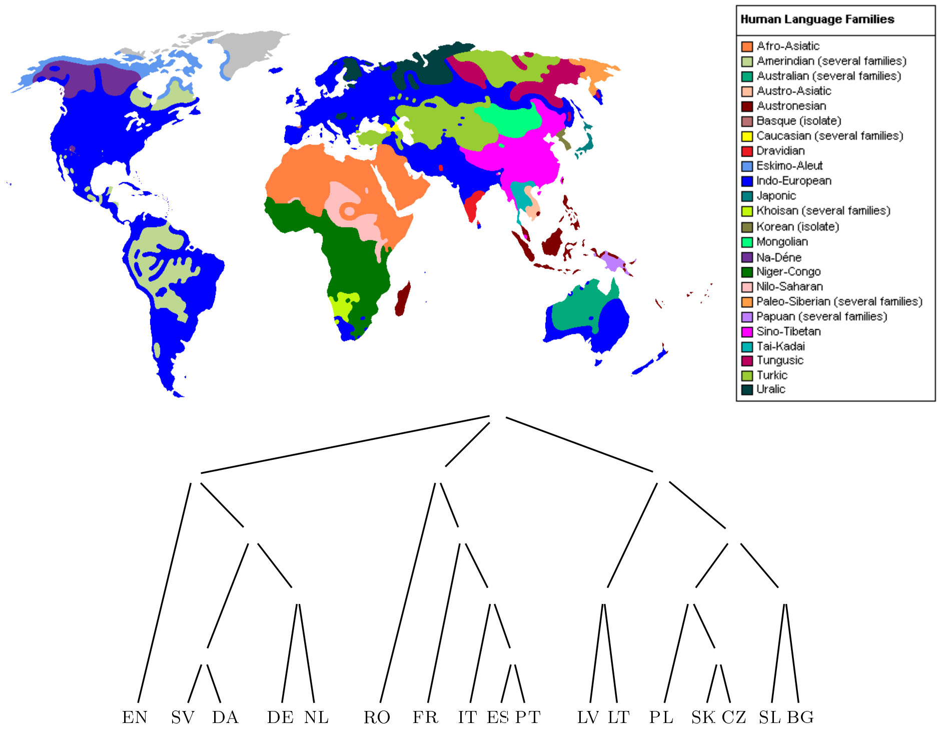

Figure 3.1: *Above*: the geographic spread of the world's language families indicated by different colours. *Below*: a phylogenetic tree for a subset of the Indo-European family. (Source of above image: Wikimedia Commons.)

computational approaches have been taken to automate some of the involved tasks [17, 21, 64, 73, 89]. The study of *etymology*, the history of words, is another branch of historical linguistics and closely related to the comparative method. To determine language relatedness, it is essential to compare the words of different languages and to identify those that share a common origin, that is, that are etymologically related, as well as the type of etymological relation.

Formally, in cognate identification we are given two string sets $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$, and the task is to extract those pairs $(x, y)$ in relation $R$ such that

$$R = \{(x, y) \in X \times Y | \ \ x \text{ is cognate with } y \ \}$$

where each element $x \in X$ and $y \in Y$ is a string over alphabets $\Sigma_x$ and $\Sigma_y$ respectively. The alphabet sets do not necessarily overlap, since the orthographies of different languages tend to vary.

Next, we clarify several kinds of cross-lingual etymological relations and give our definition of cognacy. Then, we review the literature on two kinds of approaches to the cognate identification task: heuristic string metrics and more recent data-driven methods.

### 3.1.1 Etymological Relations

In historical linguistics, *cognates* are words from different contemporary languages descending from the same historical root word. Figure 3.2 illustrates the cognate relation and an example cognate set for the word *night*. Cognates only exist between related languages, such as English and German (e.g. *head* and *Haupt*). *Loanwords*, in contrast, are words that have been borrowed from one language to another through language contact, regardless of relatedness. For example, Finnish and Swedish have many loanwords between each other despite being unrelated. Loanwords can be phonetically adapted to the borrowing language (e.g. English *mutton* from French *mouton*), borrowed directly (e.g. English *rucksack* from German *Rucksack*), or translated (e.g. English *loanword* from German *Lehnwort*). Etymological relations exist also within languages, such as derivations and compounds, but these are not relevant in cross-lingual settings.

The definition of cognate varies in NLP literature. Apart from work using the historical linguistic definition given above [62, 76, 89], more relaxed definitions have been motivated by practical concerns. Some authors refer to any etymologically related pair of words as cognates, including loanwords [10, 13, 74], while others define cognates as words sharing both a similar form and common meaning [12, 100]. This last definition is problematic for historical linguistics, since it excludes cognate words having different meanings, but may be more useful for some applications, such as

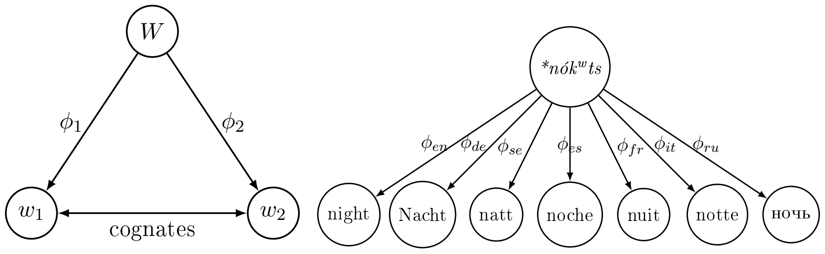

Figure 3.2: *Left:* the cognate relation, where words $w_1$ and $w_2$ from two contemporary languages originate from some common root $W$, through phonetic transformations $\phi_1$ and $\phi_2$. *Right*: contemporary cognates to the word *night* descending from a reconstructed Proto-Indo-European root.

| **Word $x$** | **Word $y$** | **Meaning of $x$** | **Meaning of $y$** |
|---|---|---|---|
| it: *notte* | es: *noche* | 'night' | 'night' |
| en: *attend* | fr: *attendre* | 'attend' | 'wait' |
| fi: *huvittava* | et: *huvitav* | 'amusing' | 'interesting' |
| en: *oath* | sv: *ed* | 'oath' | 'oath' |
| fi: *pöytä* | sv: *bord* | 'table' | 'table' |
| en: *bite* | fr: *fendre* | 'bite' | 'split' |

Table 3.1: Examples of etymologically related words (i.e. cognates). The degree of similarity in form and meaning varies significantly.

word translation [53]. In this thesis, we regard any cross-lingual pair of etymologically related words as cognates, including both true cognates as well as loans from one language to another.

Several factors have been found to predict cognacy: phonetic similarity, reflected by orthographic similarity [16], semantic similarity, and the presence of *regular sound correspondences*, word segments regularly occurring in similar phonetic positions and contexts [76]. The example cognates in Table 3.1 illustrate the complexity of cognate identification. A straightforward example is the Italian–Spanish pair *notte–noche*, with a similar form and common meaning. However, many cognates have similar surface forms, but differ in meaning, such as the English–French *actual–actuel* and Finnish–Estonian *huvittava–huvitav*, that is, they are *false friends*.

Furthermore, cognates might look very different on the surface. English–Swedish cognates *oath–ed* and Finnish–Swedish *pöytä–bord* look quite different, but share a meaning. On the other hand, English–French *bite–fendre* are similar neither in form nor meaning. The only way to recognise such cognates from their surface forms alone is to identify regular correspondences, such as *th–d* for English–Swedish. Consequently, and in contrast to much previous work, we make no strict assumptions about the degree of similarity in form or meaning that any two cognates should exhibit. Instead, following Jäger [64], we treat regular correspondences as the main driving factor in the cognate relation and attempt to capture these in a completely data-driven manner.

### 3.1.2 String Metrics

Earlier computational approaches to cognate identification attempt to design a string similarity (or distance) metric that assigns a higher score to cognate words and a lower score to unrelated ones. A common approach is to extend the traditional *Levenshtein distance* [85], also known as *edit* *distance.* The Levenshtein distance (LD) between strings $s_1$ and $s_2$ over an alphabet $\Sigma$ is computed as the minimum number of insertion, deletion, or substitution operations needed to transform one string to the other. For example, cognate pairs *coupe–Kopf* and *pöytä–bord* have distance values of 3 and 5. That is, *coupe* can be transformed into *Kopf* with three operations, namely substitutions *c→K* and *e→f*, and deletion of *u*. Likewise, with four substitutions (*p→b, ö→o, y→r, t→d*) and one deletion (of *ä*), *pöytä* is transformed into *bord.*

LD has been extended by associating specific weights to pairs of symbols using linguistic knowledge [e.g. 73, 89], or sets of example cognates [e.g. 12, 117]. For example, the ALINE algorithm [73] uses specific weights based on several pre-determined phonetic features. Furthermore, LD has been generalised with an $n$-gram similarity measure [75]. Another method involves a heuristic based on mapping consonants to ten classes, and consider words matching in their first two consonant classes to be cognates [141]. Similarly, the sound-class-alignment algorithm of List [89] uses a large set of sound classes and also considers prosodic aspects of words. Others rely on learning regular correspondences (sometimes called *mismatches* or *substitution patterns*) from example cognates using an alignment algorithm. For example, Gomes and Pereira Lopes [41] develop a weighted string similarity metric for words in orthographic form by extracting substring pairs from example cognate sets.

### 3.1.3 Data-Driven Approaches

In addition to weighted string similarity metrics, especially more recent work on cognate identification has utilised data-driven approaches, most notably the support vector machine [29]. The support vector machine (SVM) is a supervised learning model trained by finding the optimal separating hyperplane between multi-dimensional data points of different classes.

The basic SVM is a non-probabilistic, linear binary classifier. For data that is linearly separable, the optimal hyperplane creates the maximum margin between training points in the two classes. The maximum-margin hyperplane is of the form

$$\mathbf{w}^T\mathbf{x}_i + b = 0$$

where $\mathbf{x}_i$ is an input vector, $\mathbf{w}$ is a weight vector, and $b$ is a bias term. The classes are then separable so that $\mathbf{w}^T\mathbf{x}_i + b < 0$ when the corresponding class is $y_i = 0$, and $\mathbf{w}^T\mathbf{x}_i + b \geq 0$ when $y_i = 1$. When the data classes are not linearly separable, the margin can still be maximised while allowing some data points to be on the wrong side of the optimal hyperplane. Another approach is to use a non-linear kernel function, such as a polynomial or a radial-basis function, to transform the feature space to a higher-dimensional one, making the classes linearly separable.

In previous work, SVMs have been used in combination with string similarity metrics. For examples, substring pairs [12, 27] or gap-weighted subsequences extracted from cognate sets have been used as SVM features. Similarly, Hauer and Kondrak [52] convert word pairs into features for an SVM using a set of string similarity metrics. This approach has been extended with features for semantic similarity, for example using the lexical database WordNet [62, 76, 136]. The use of global constraints and reranking [13] has also improved the performance of SVMs relying on string-similarity features. In Chapter 4, we use a linear SVM, vectorising word pairs with string metrics.

Pre-trained language representations have also been used to enhance cognate identification, including both cross-lingual word embeddings [66, 136] and character embeddings for a weighted LD-based metric [94]. A similar approach to learning weights is by pointwise mutual information [64]. More recently, also neural networks have been applied to cognate identification. Rama [118] proposes a Siamese convolutional network to identify cognates in multilingual wordlists, using both phonetic and one-hot vectors as input encodings. We propose a similar neural model in Chapter 4. On the other hand, Kanojia et al. [67] use a Siamese feed-forward network. Furthermore, Hämäläinen and Rueter [50] apply neural machine translation to cognate prediction, similarly to Beinborn at al. [10].

## 3.2 Document Planning

*Document planning* is a fundamental task in data-to-text natural language generation (NLG), the production of text outputs from underlying data in non-linguistic form [121]. According to Gatt and Krahmer [37], data-to-text NLG consists of six main stages, forming a pipeline turning input data into output text. The stages include the following:

(1) *Content determination*, selecting the information in the output;

(2) *Text structuring*, determining the order of presented information;

(3) *Sentence aggregation*, allocating information to individual sentences;

(4) *Lexicalisation*, finding words and phrases to express the information;

(5) *Referring expression generation*, selecting words and phrases to identify different entities; and

(6) *Linguistic realisation*, combining words and phrases into well-formed sentences.

Document planning consists of the first two stages, content determination and text structuring [137]. In this thesis, we refer to these stages as *content selection* and *document structuring*, following Leppänen and Toivonen [84]. These stages are usually preceded by turning the data into *messages*, semantic representations of input data samples. In Chapter 6, we consider document planning in the context of their system, generating statistical news reports from structured data in a truly low-resource scenario. Similar resource scarcity is common in NLG tasks [57].

### 3.2.1 Characteristics of News

The reporting of *news* as a presentation of observed facts is a common application of data-to-text NLG [37]. Such news have been referred to as *hard news* [84], including domains such as sports, weather, finance, economy, and so on. In Chapter 6, we deal with the generation of statistical news based on consumer price index data.

Since data usually contains more information than is desirable to be included in the output, content selection attempts to filter out the most relevant information from input data. While the appropriate method depends on the domain, one way is to use measures of statistical outlierness [38, 83], which we consider applicable to our statistical news context. Intuitively, newsworthy information tends to deviate from the ordinary.

Regarding the structure of news, the most important information tends to be presented at the beginning, followed by content providing additional information [138]. This concept of importance has been used in previous work on document planning [8, 38]. Furthermore, hard news have also been characterised by an orbital structure, where a *nucleus* (e.g. first paragraph) is 'orbited' by *satellites* (other paragraphs) [148]. While the nucleus presents the most important information, satellites provide additional details about the nucleus. A similar terminology is used also in rhetorical structure theory [93] to describe relations within paragraphs, suggesting that the orbital theory can be applied to sentences within paragraphs [84].

### 3.2.2 Heuristics and Neural Methods

Despite being a crucial task for generation quality, document planning has been relatively little addressed in NLP research [137]. Hand-engineered approaches to document planning are based on rules or templates [37], or specific metadata [84]. Although they are often more reliable than neural approaches [154], the scalability of these methods across domains is limited by their dependence on domain-specific knowledge and heuristics.

More recently, an increasing amount of research has investigated data-driven, and especially neural, approaches to the task, thus forgoing hand-engineered rules. For example, Puduppully et al. [114] train a sequence-to-sequence network with attention end-to-end, while modelling document planning explicitly. Although such approaches are promising, they usually depend on annotated training datasets where input data and human-written output texts are aligned [5, 114, 115, 150]. Since such datasets are expensive to gather, many domains are truly low-resource with regard to this task even in generally high-resource languages [57, 137]. Addressing this issue of data availability, Laha et al. [77] convert structured data first into tuples and then into simple sentences using a sequence-to-sequence network. However, they do not conduct any reordering of sentences, reducing the cross-domain applicability of their method. In this thesis, we propose a neural approach using distant supervision in Chapter 6.

### 3.2.3 Sentence Ordering

In Chapter 6, we propose distant supervision for document planning by pre-training a neural model for sentence ordering, our source task. As illustrated in Figure 3.3, in sentence ordering we are given an unordered set of sentences $S = \{s_1, \ldots, s_n\}$, and our goal is to predict its correct ordering $O = (o_1, \ldots, o_n)$, thus producing the correct sentence sequence $S' = (s_{o_1}, \ldots, s_{o_n})$. Sentence ordering is a commonly used task in coherence modelling, and a model trained for the task can be useful for many other NLP tasks in addition to text generation, such as summarization, question answering, and so on. As the aim of document planning is to produce a coherent presentation of relevant information, we consider a sentence ordering an appropriate source task. In previous work, several kinds of machine-learning-based approaches have been taken to address the sentence ordering problem, most of the recent work using neural networks. These approaches include pairwise approaches [2, 24, 113, 128], graph-based models [86, 155], sentence position modeling [14], and sequence-to-sequence models [42, 91, 102, 147].

In pairwise approaches, a model is trained to predict whether one sentence should precede another. Such a model can then be used for sorting an unordered set of sentences by pairwise comparisons. This introduces a combinatorial search problem, which has been addressed using beam search [24] and other approximations [2], as well as topological sort [113, 128]. The models used include Siamese neural networks with convolutional [24] or re-

```
(1) In contrast, wholesale trade sales went down by 1.5 per cent and
    motor vehicle trade sales by 27.3 per cent from twelve months back.
(2) The volume of sales, from which the impact of prices has been
    eliminated, grew by 1.4 per cent in retail trade over the same
    period.
(3) According to Statistics Finland, retail trade sales grew by 4.6 per
    cent in November from November 2011.
(4) In total trade, sales contracted by 3.4 per cent in November.
(5) In daily consumer goods trade, sales grew by 5.4 per cent and in
    department store trade by 4.6 per cent year-on-year.
```

$$\Downarrow$$

```
(3) According to Statistics Finland, retail trade sales grew by 4.6 per
cent in November from November 2011. (2) The volume of sales, from which
the impact of prices has been eliminated, grew by 1.4 per cent in retail
trade over the same period. (5) In daily consumer goods trade, sales grew
by 5.4 per cent and in department store trade by 4.6 per cent year-on-year.
(1) In contrast, wholesale trade sales went down by 1.5 per cent and motor
vehicle trade sales by 27.3 per cent from twelve months back. (4) In total
trade, sales contracted by 3.4 per cent in November.
```

Figure 3.3: The sentence ordering task.

current layers [2, 113]. In contrast, a sentence position model is trained on individual sentences instead of pairs [14]. This approach is based on predicting which quantile of the correct sentence sequence (e.g. paragraph) a sentence should belong to. Using such predictions, a sentence can be sorted according to the average quantile predicted by the model.

More recent sequence-to-sequence models are mostly based on the pointer network architecture of Vinyals et al. [145]. For example, some work employs recurrent neural networks to encode the sentence set $S$ into a paragraph vector, with a pointer network then predicting the correct sequence $S'$ using attention to 'point' to input sentences $s_i$ [42, 91]. This approach has been further developed with a hierarchy of recurrent attention-based word and sentence encoders [147]. Another lines of work combine a pointer-network approach with topic models [102], or employ graph RNNs creating graph representations based on entities shared between sentences [156]. More recently, the pointer-network approach has been further extended using Transformer-based (see Section 2.1.5) language models, for example BERT [32], to obtain better paragraph representations [128].

# Chapter 4

# Cognate Identification in Sami Languages

In this chapter, we examine the problem of identifying etymologically related words, *cognates*, within a set of endangered Sami languages of the Uralic family. As stated earlier in Chapter 1, our aim in this chapter is to address our research task RT-I:

**RT-I.** *Given scarce language-specific labelled data, identify cognates in truly low-resource Sami languages.*

To accomplish RT-I, our approach is *similarity learning*, that is, to learn a similarity metric from data that can predict the cognacy for a given cross-lingual word pair. To do this, we propose two models: a support vector machine (SVM), and a multi-input, Siamese convolutional neural network (S-CNN). While the SVM has been a popular method for cognate identification [52, 117], neural networks represent a more recent development [67, 118]. In contrast to much of earlier work relying on linguistic resources [e.g. 136], we propose models that are language-independent in order to promote easier scalability to new languages.

Our labelled target data consists of a small cognate set for North, South, and Skolt Sami, which is insufficient to train our models for similarity learning. Hence, our approach is to pre-train our models on language pairs of the high-resource Indo-European family, and transfer these models to our Sami set. That is, whereas most previous work addresses transfer between closely related languages [e.g. 94], we consider a setting where they are highly unrelated. Since cognacy is driven by phonetic changes that are more universal than languages in general (see background in Section 3.1), we hypothesise that patterns in cognacy can carry over across language families.

In this chapter, we conduct two experiments. First, in Section 4.3, our aim is to test our similarity learning approach for cognate identification, and to find out whether pre-training on unrelated language data is beneficial. Second, in Section 4.4, we fine-tune the neural model with a portion of the Sami cognate set, to quantify the benefit gained from having access to small amounts of labelled Sami data, or conversely, the loss in performance caused by lack of such data.

In the next section, we define our problem setting more formally through the transfer learning framework presented in Section 2.2. Then, in Section 4.2 we present our methods in detail. In addition to our similarity learning models, we present string metrics that we use as the SVM's features, of which one we use as a baseline. In Sections 4.3–4.4, we present our two experiments: the datasets used for training (or fine-tuning) and testing, our implementations, evaluation schemes, and results. Finally, we conclude the chapter with a discussion on the findings.

## 4.1 Problem

In this chapter, our aim is to identify *cognates* within a truly low-resource language group using data from a high-resourced language family. As discussed in Section 3.1, the term cognate has several uses in literature. Following previous work [10, 13, 74], we regard any pair of etymologically related words as cognates, regardless of the type of etymological relation.

Our datasets consist of etymological data for two, highly unrelated language families, where one family contains several high-resourced languages and the other extremely low-resource ones. These datasets contain positive labels only (i.e. verified cognate pairs). In our transfer learning setting, these language families represent the source and target domains $\mathcal{D}_S$ and $\mathcal{D}_T$, where $\mathcal{D}_S$ is the high-resourced Indo-European language family and $\mathcal{D}_T$ is the truly low-resource Sami language group of the Uralic language family.

Following the definitions and notation of Ruder [123] (see Section 2.2), we define the source and target domains as pairs of feature spaces and marginal distributions

$$\begin{cases} \mathcal{D}_S = \{\mathcal{X}_S, P_S(X_S)\} \\ \mathcal{D}_T = \{\mathcal{X}_T, P_T(X_T)\} \end{cases}$$

where $\mathcal{X}_S, \mathcal{X}_T$ are the source and target feature spaces and $P_S(X_S), P_T(X_T)$ their respective marginal distributions, where $X_S, X_T$ are random variables associated with the word pair samples observed in source and target datasets. We define the corresponding tasks as triples of label spaces, prior and conditional distributions

$$\begin{cases} \mathcal{T}_S = \{\mathcal{Y}_S, P_S(Y_S), P_S(Y_S|X_S)\} \\ \mathcal{T}_T = \{\mathcal{Y}_T, P_T(Y_T), P_T(Y_T|X_T)\}, \end{cases}$$

where $\mathcal{Y}_S, \mathcal{Y}_T$ are the label spaces and $P_S(Y_S), P_T(Y_T)$ their respective prior distributions, $P_S(Y_S|X_S), P_T(Y_T|X_T)$ are the conditional distributions learned from data, and $Y_S, Y_T$ are random variables.

In the setting of this chapter, the feature spaces $\mathcal{X}_S$ and $\mathcal{X}_T$ are all the distinct cross-lingual pairings of words within the source and target language families $\mathcal{F}_S$ and $\mathcal{F}_T$, which represent the Indo-European and Uralic families. That is, a feature space $\mathcal{X}$ is of the form

$$\mathcal{X} = \{(w^{\mathcal{L}_i}, u^{\mathcal{L}_j}) \in [\mathcal{F}]^2 \ \mid \ i \neq j\}$$

where $\mathcal{L}$ is a language, words $w, u$ are strings over language-specific alphabets $\Sigma_{\mathcal{L}}$, $\mathcal{F}$ is a language family, and $[\mathcal{F}]^2$ denotes the set of all word pairs within $\mathcal{F}$. We define language $\mathcal{L}$ as a set of words, and a family $\mathcal{F}$ as the union of such language-specific word sets, that is

$$\mathcal{F} = \{\mathcal{L}_1 \cup \cdots \cup \mathcal{L}_N\} = \{w_1^{\mathcal{L}_1}, \ldots, w_{n_1}^{\mathcal{L}_1}, w_1^{\mathcal{L}_2}, \ldots, w_{n_2}^{\mathcal{L}_2}, \ldots, w_1^{\mathcal{L}_N}, \ldots, w_{n_N}^{\mathcal{L}_N}\}$$

where $n_i = |\mathcal{L}_i|$ and $N$ is the number of languages. While the families $\mathcal{F}_S$ and $\mathcal{F}_T$ are distinct sets of languages, that is $\mathcal{F}_S \cap \mathcal{F}_T = \emptyset$, their alphabet sets $\Sigma_{\mathcal{F}_S}$ and $\Sigma_{\mathcal{F}_T}$ may overlap. In our case, these are the sets of Unicode characters occurring in the words of each family. As noted in Section 3.1, it is likely that there is more alphabet overlap within the families than between them.

We formulate both the source and target tasks $\mathcal{T}_S, \mathcal{T}_T$ as binary classification. Thus, the label spaces are simply $\mathcal{Y}_S = \mathcal{Y}_T = \{0, 1\}$, one referring to the positive class where two words are cognates. The source and target datasets $D_S, D_T$ are samples of cross-lingual word pairs from feature spaces $\mathcal{X}_S$ and $\mathcal{X}_T$ associated with binary labels, although $D_T$ is only partially labelled. That is, these datasets are of the form

$$\begin{cases} D_S = \{(x_i^S, y_i^S)\}_{i=1}^{N_{D_S}} \\ D_T = \{(x_i^T, y_i^T)\}_{i=1}^{p} \cup \{x_i^T\}_{p+1}^{N_{D_T}} \end{cases}$$

where $x_i = (x_{i1}, x_{i2}) \in \mathcal{X}$ are the cross-lingual word pair samples, $y_i \in \mathcal{Y}$ are the associated binary labels, and $N_{D_S}, N_{D_T}$ are the sizes of the datasets. The source dataset $D_S$ is fully labelled such that $\forall i \quad y_i^S = 1$, that is, it contains only positive samples of cognate pairs. The target dataset $D_T$ contains a small amount $p$ of labelled samples, again positive only, that is, $y_i^T = 1$ when $1 \leq i \leq p$. The rest of the samples in $D_T$ are unlabelled. Since the samples of $D_S$ and $D_T$ come from their respective families $\mathcal{F}_S$ and $\mathcal{F}_T$, it is extremely unlikely they would have any word pairs in common.

Referring to the transfer learning scenarios described in Section 2.2.2, we are facing scenarios 4 and 5 [123]. That is, $\mathcal{X}_S \neq \mathcal{X}_T$, since the word pairs observed in $D_S$ and $D_T$ are very unlikely to overlap. Also, since there is no one-to-one mapping between the samples, it must be that $P_S(X_S) \neq P_T(X_T)$. Thus, we regard our setting as a combination of domain adaptation and cross-lingual learning (see background in Section 2.2), where we have a small amount of labelled target data available.

Given these source and target datasets $D_S$ and $D_T$, we want to identify those word pairs $x^T = (x_1^T, x_2^T) \in D_T$ where $x_1^T$ and $x_2^T$ are cognate to each other. As cognacy is difficult to define in terms of rules for computational purposes, and as we are interested in language-independent methods not relying on linguistic knowledge, we consider a data-driven approach appropriate to this task. We formulate our task as the classification of cross-lingual word pairs into cognate and non-cognate pairs such that the prediction for some word pair $(x_1^T, x_2^T)$ being cognate is $\hat{y}^T = p(y^T = 1 \mid (x_1^T, x_2^T)) \in [0, 1]$. Since the input consists of word pairs and the output is between zero and one, such a model can be considered to learn and predict the *similarity* (in terms of cognacy) between any two words from different languages.

Thus, our objective is to optimise for the parameters $\theta$ of the model $p(y^T \mid (x_1^T, x_2^T), \theta)$ for the classification of word pairs in the unlabelled target data $D_T$. Since only few labels are available for $D_T$, while $D_S$ is thoroughly labelled with positive examples only, our approach is to first build a training dataset, *pre-train* the model on $D_S$, and then *fine-tune* the model on a dataset built from the $p$ labelled examples from $D_T$. Since there is practically no overlap between $D_S$ and $D_T$ with regard to their sample word pairs, but there may be between their alphabet sets, our strategy is to train a model for learning patterns between cross-lingual pairs of *substrings*, contiguous sequences of characters.

## 4.2 Methods

In this section, we present three methods for cognate identification: string metrics, a support vector machine (SVM), and a Siamese convolutional neural network (S-CNN). Each of these three methods represents a different approach to the problem, and especially the first two have been used extensively in previous work. String metrics are the algorithmic, more traditional tool of choice [12, 73], while SVM is a supervised learning model found useful for the task [62, 136]. The S-CNN, in turn, represents a newer line of work using neural networks, which have been successful in many other NLP tasks [67, 118]. Whereas the following string metrics are directly from previous work, the SVM and S-CNN are our modifications of similar models, omitting all language-specific features, following our premise of language-independence.

## 4.2.1 String Metrics

We present three string metrics used in our experiments: Levenshtein-distance similarity (LDsim), number of common bigrams (ncb), and prefix length (pref). LDsim is our baseline, while the other two metrics we use as the SVM's features (see Section 4.2.2).

Although many other string similarity metrics exist, as discussed in Section 3.1.2, we consider LDsim a suitable baseline as a straightforward and general metric for string similarity. We also consider common bigrams and prefix length to be suitable metrics for cognate identification, as we aim to identify patterns between pairs of substrings, and since words sharing their first letters are more likely to be cognate [136]. In addition, some of the other common metrics (e.g. longest common subsequence, common $n$-grams) have been found redundant to these three [52].

The *Levenshtein distance* [85], or *edit distance*, between strings $s_1$ and $s_2$ over an alphabet $\Sigma$ is the minimum number of insertion, deletion, or substitution operations needed to transform one string to the other, as explained earlier in Section 3.1.2. To obtain the *normalised* Levenshtein distance, this number is divided by the length of the longer word, equal to the maximum possible distance between $s_1$ and $s_2$. The similarity metric is then defined as

$$\text{LD}\textsc{sim} = 1 - \frac{d_L(s_1, s_2)}{max\{|s_1|, |s_2|\}} \in [0, 1],$$

where $d_L(s_1, s_2)$ is the Levenshtein distance. For example, for cognate pairs *coupe–Kopf* and *pöytä–bord*, the respective similarities are $1 - \frac{3}{5} = 0.4$ and $1 - \frac{5}{5} = 0$. This method is most suitable when the alphabets are shared between languages, since the similarity is always zero for disjoint alphabets. (The same applies to our other string metrics as well.)

One of the metrics we use in our SVM model is the normalised number of common bigrams between strings $s_1$ and $s_2$, which we refer to as $\textsc{ncb}(s_1, s_2)$. A *bigram* of a string $s$ is any two adjacent elements within $s$, in other words any of its substrings of length two. For example, if $s_1 = \textit{huvittava}$ and $s_2 = \textit{huvitav}$, their bigram sets are $B(s_1) = \{\textit{hu, uv, vi, it, tt, ta, av, va}\}$ and $B(s_2) = \{\textit{hu, uv, vi, it, ta, av}\}$, and their normalised number of common bigrams is

$$\textsc{ncb}(s_1, s_2) = \frac{|B(s_1) \cap B(s_2)|}{\max\{|B(s_1)|, |B(s_2)|\}} = \frac{|B(\textit{huvittava}) \cap B(\textit{huvitav})|}{\max\{|B(\textit{huvittava})|, |B(\textit{huvitav})|\}}$$
$$= \frac{6}{8} = 0.75.$$

The third metric, the length of the longest common prefix between $s_1$ and $s_2$, is simply the length of the longest common substring starting from the first characters of $s_1$ and $s_2$. For example, for strings $s_1 = \textit{notte}$ and $s_2 = \textit{noche}$ this value is 2. To obtain a normalised value, we divide this value by the length of the shorter word, so that a value of one means the shorter word is a prefix of the other word and zero that the words have no common prefix. We refer to this metric as $\textsc{pref}(s_1, s_2)$.

### 4.2.2 Support Vector Machine

Our SVM model (see Section 3.1.3) uses word similarity metrics as features for word pairs. The features we use are LDsim, ncb, pref, normalised lengths of both words, and the normalised absolute difference between the lengths. That is, we represent a word (string) pair $(s_1, s_2)$ as a feature vector $\mathbf{x} \in \mathbb{R}^6$ such that

- $x_1 = \text{LD}\textsc{sim}(s_1, s_2)$,
- $x_2 = \textsc{ncb}(s_1, s_2)$,
- $x_3 = \textsc{pref}(s_1, s_2)$,
- $x_4 = \frac{|s_1|}{\max\{|s_1|,|s_2|\}}$,
- $x_5 = \frac{|s_2|}{\max\{|s_1|,|s_2|\}}$, and
- $x_6 = \frac{||s_1|-|s_2||}{\max\{|s_1|,|s_2|\}} = \frac{|x_4-x_5|}{\max\{|s_1|,|s_2|\}}$.

These features are similar to those of Hauer and Kondrak [52], but in contrast to their work we use normalised values and LDsim instead of the Levenshtein distance, and omit their language-pair features. Otherwise, we use the same metrics apart from LDsim, since they have been found to produce good results [52].

We opt to use LDsim instead of absolute Levenshtein distance, as its values are normalised to the same scale as the other features, which should be better for SVM performance. We use normalised values of all features in order to diminish the effect of word length on classification, as we do not want the model to disregard cognates with greater length. We normalise the values by dividing them with the length of the longer word, except for PREF, which divides with the length of the shorter word. We omit the language-pair features as they are not applicable to our language-independent setting. We do not use any phonetic similarity metrics, as our datasets contain orthographic forms only. It is also more likely the case with truly low-resource languages that more data is available in orthographic than in phonetic form.

We have chosen this SVM model as it is based on such string similarity measures that are applicable to our low-resource scenario. More advanced SVM-based approaches to cognate identification exist, but they either require detailed dictionary definitions in a high-resource language with high-quality pre-trained word embeddings [136], or multilingual wordlists aligned by concepts [62], which we do not have.

### 4.2.3 Siamese Convolutional Neural Network

As our neural method for cognate identification, we use a Siamese convolutional neural network (S-CNN, see Section 2.1.4). A similar model was first used for cognate identification by [118], modifying the original Siamese network developed for the task of face verification [26]. Our implementation is similar, although we omit their language features and phonetic representations for the same reasons as with the SVM. The architecture we use is illustrated by Figure 4.1.

Since the CNN requires a grid-like representation of the input (see Section 2.1.3), we represent a word as a matrix $\mathbf{X} \in \{0,1\}^{|\Sigma| \times n}$ such that $\mathbf{X} = [\mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_n]$, where each column vector $\mathbf{x}_i \in \{0,1\}^{|\Sigma|}$ is a one-hot vector representing a character in the alphabet $\Sigma$. The training data $D = \{(\mathbf{X}_{ai}, \mathbf{X}_{bi}), y_i\}_{i=1}^{N}$ then consists of pairs of words such that $y_i = 1$ if $\mathbf{X}_{ai}$ and $\mathbf{X}_{bi}$ are cognates, and $y_i = 0$ otherwise.

As shown in Figure 4.1, the S-CNN model is an extension of the CNN: first, a set of filters $\mathbf{W} \in \mathbb{R}^{p \times q}$ are convolved (cross-correlated) over character sequences of a pre-determined length $q$ from both input matrices $\mathbf{X}_a$ and $\mathbf{X}_b$, each filter producing a feature map for each input. That is, one filter is applied to both inputs, meaning that the weights are tied and the two CNNs are identical functions mapping similar words close to each other in the representation space. Siamese networks have been found to achieve better performance with tied weights [72].

To the feature maps, we then apply a rectified linear unit (ReLU) activation function and max-pooling. The number of feature maps produced

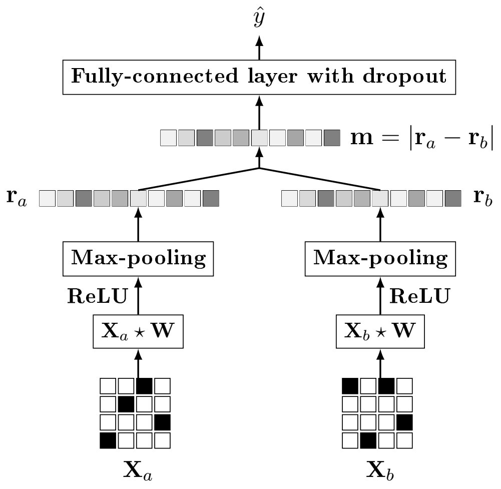
$\hat{y}$
Fully-connected layer with dropout
$\mathbf{m} = |\mathbf{r}_a - \mathbf{r}_b|$
$\mathbf{r}_a$
$\mathbf{r}_b$
Max-pooling
Max-pooling
ReLU
ReLU
$\mathbf{X}_a \star \mathbf{W}$
$\mathbf{X}_b \star \mathbf{W}$
$\mathbf{X}_a$
$\mathbf{X}_b$

Figure 4.1: Architecture of the S-CNN. Column vectors in input matrices represent one-hot-encoded characters. The same filter $\mathbf{W}$ is convolved with both inputs, i.e. the weights are tied.

from each input matrix is equal to the number of filters. We fix the filter height at $p = |\Sigma|$, equal to the size of the alphabet and the height of the input matrix, resulting in vector-formed feature maps. We choose to use the ReLU activation and max-pooling, as they have been found effective in CNNs for NLP [70, 118].

We obtain the representation vectors $\mathbf{r}_a$ and $\mathbf{r}_b$ by concatenating all the feature map vectors into single vectors. These two vectors are then merged into one vector $\mathbf{m}$ using some distance metric. We use the absolute vector difference [118] such that $\mathbf{m} = |\mathbf{r}_a - \mathbf{r}_b| = [|r_{a1} - r_{b1}|, \ldots, |r_{al} - r_{bl}|]^T$, where $l = |\mathbf{r}_a| = |\mathbf{r}_b|$. Finally, the merged vector $\mathbf{m}$ is fed as input to a fully-connected layer, itself connected to the output neuron. The dropout technique [135] is applied to the fully-connected layer for regularisation, and the output neuron is activated with the sigmoid function to produce a prediction $\hat{y} \in [0, 1]$. The output of a trained model can be regarded as a learned similarity metric between pairs of inputs, although it does not satisfy the triangle inequality as true metrics do.

## 4.3 Indo-European Models for Sami Cognates

In this section, we present our first experiment, where we train the SVM and S-CNN models on Indo-European data and directly apply them to cognate identification between North, South, and Skolt Sami of the Uralic language family. As our baseline we use LDSIM, which as a string metric requires no training. Next, we describe our datasets, implementation, evaluation metrics, and results.

### 4.3.1 Datasets

In order to train the SVM and S-CNN models, we build a training dataset from the Etymological WordNet [39]. This is a database containing information of etymological origin, cognacy, as well as derivational and compositional links between words (see Section 3.1.1 for definitions). It consists of word pairs that each belong to one of the aforementioned relations. The database has been mined from Wiktionary, and its entries are mostly from widely spoken Indo-European languages, such as English, German, French, and so on, but also from historical languages such as Latin and Old English.

To build our training dataset, we include all etymologically related word pairs where the two words either share a common root or one word is the root of the other. We disregard all derivationally or compositionally linked word pairs, as such links are not relevant to cognacy. Furthermore, we filter out those pairs where both words belong to the same language, as we are concerned with identifying cognates across languages. In total, there are 73,238 cognate pairs in the filtered training set.

As our models are binary classifiers, our training dataset requires also negative examples of non-cognate word pairs. Thus, we build the training dataset from the cognate set by pairing unrelated words randomly, so that 10% of the pairs are cognate. We consider this a suitable ratio, as only a tiny amount of word pairs out of all possible pairs between two languages are cognate, depending on the relatedness of the languages. On the one hand, the more balanced the classes are, the more false positives will be predicted by the models, but on the other hand, a too small portion of cognate pairs could result in more false negatives and the models not learning anything about cognacy. While our chosen proportion is unrealistically high, we regard the cost of misclassifying cognates as unrelated (false negatives) higher than that of misclassifying unrelated words as cognates (false positives). We refer to our Indo-European training set as IE-TRAIN.

To test our models, we use a set of three vocabularies from North, South, and Skolt Sami of the Uralic language family. We retrieved these vocabularies from dictionaries compiled by Giellatekno[1]. In order to build our test dataset, we filtered out all words with upper-case (proper nouns)

[1]The research group of Sami language technology at the University of Tromssa. `http://giellatekno.uit.no/index.eng.html`.

| Family | Dataset | | #pairs | #cognates | $|\Sigma|$ |
|---|---|---|---|---|---|
| Indo-European | IE-Train | | 732,380 | (73,238) | 329 |
| Uralic | Sami-Full | sma–sme | 11,234 × 47,312 | (1,460) | 42 (27) |
| | | sma–sms | 11,234 × 29,401 | (838) | 75 (27) |
| | | sme–sms | 47,312 × 29,401 | (2,188) | 77 (38) |

Table 4.1: Our Indo-European and Sami datasets used for pre-training and testing, respectively. Languages contained in Sami-Full are South Sami (sma), North Sami (sme), and Skolt Sami (sms). $|\Sigma|$ is the alphabet size, i.e. total number of symbols observed in a dataset, the number of overlapping characters given in parentheses for the language pairs of Sami-Full.

or non-alphabetic characters, and obtained gold-standard cognate sets for these languages from Álgu[2], an etymological database for Sami languages. This database contains (positive-only) cognate information for only a subset of all the words in the vocabularies, meaning that our test dataset is only partially labelled and for positive examples only. That is, some of the unlabelled word pairs might in fact be cognate, although the vast majority of them are unrelated. We refer to our test dataset as Sami-Full and average results over the three pairs of languages. Statistics for both IE-Train and Sami-Full are shown in Table 4.1, including the number of cognates and total word pairs, and alphabet sizes. Despite relatively close relatedness between the languages of Sami-Full, we note that the alphabets of the three Sami languages are not very overlapping, which makes the cognate identification task more difficult.

## 4.3.2 Implementation

In our implementation of the S-CNN model, we use ten filters with height $p = |\Sigma|$ (alphabet size) and width $q = 2$. The alphabet is the set of all characters observed in both the training and test datasets, and its size is $|\Sigma| = 336$. We fix the input matrix width to $n = 20$. For words shorter than this, the input matrices are zero-padded, and longer words are truncated at this length. In the fully-connected layer, we use a dropout rate of 0.5.

We train the S-CNN model using binary cross-entropy as the loss function, and the Adadelta optimizer [157] with initial learning rate $\alpha = 1.0$, decay rate $\rho = 0.95$, and the constant $\epsilon = 1 \cdot 10^{-6}$. The batch size is set to 128, and number of epochs to 50. We implemented the model using the

[2]Available at: `http://kaino.kotus.fi/algu/`

Keras library with Tensorflow backend [25]. For the SVM implementation, we use the SVM module of the Scikit-learn library for Python [105], based on the $C$-support vector classification implementation of [20]. We train the model using a linear kernel and regularisation parameter $C = 1$. The model uses Platt scaling [112] to make probabilistic predictions.

### 4.3.3 Evaluation

A difficulty in evaluating on the Sami datasets is that the set of word pairs annotated as cognates in the Álgu database is known to be far from complete for the vocabularies covered. As a result, there are many word pairs in the vocabularies that are cognates, but are evaluated as unrelated. Measures such as accuracy and precision are therefore not useful for this experiment, since we do not know whether a given word pair *not* among the annotated cognates is a cognate pair. We can, however, evaluate the *recall* of the known cognate pairs: what proportion of the annotated pairs make it into the set ranked as most likely cognates by the models.

We avoid the use of precision in order not to diminish the meaning of the results in any way. Furthermore, since the focus of this experiment is on the *identification* of cognates, we are more interested in recall than precision. However, in our second experiment we make the assumption that all word pairs except the known cognates are indeed unrelated, which is true for most word pairs in SAMI-FULL. This allows us to compute precision, and to compare models more in terms of their performance at the task, and not only whether they can identify cognates.

Applying our models to all pairs of words between two vocabularies is time consuming. Therefore, when evaluating on the language pairs of SAMI-FULL (see Table 4.1), we take from one vocabulary $A$ only those words $C_{AB} \subset A$ that we know to have at least one cognate in the other vocabulary $B$. Then, for each word $c \in C_{AB}$, we use our models to make a prediction on each word pair $(c, b)$ where $c \in C_{AB}$ and $b \in B$.

As the models' predictions are values between zero and one, we can rank the words $b \in B$ according to the likelihood of being cognate to $c \in C_{AB}$. We can then compute the recall for a set of top $k$ results averaged over both the words $c$ and the set of language pairs (including inversions, i.e. each pair is counted twice). We refer to this metric as the *mean average recall@k* (MAR@$k$), which we compute as

$$\mathrm{MAR@}k = \frac{1}{|L|} \sum_{(A,B) \in L} \frac{1}{|C_{AB}|} \sum_{c \in C_{AB}} \mathrm{R@}k(c),$$

$$\text{where } \mathrm{R@}k(c) = \frac{|\{b \in B \mid \mathrm{rank}(b) \leq k \text{ and } b \text{ is cognate with } c\}|}{|C_{BA}|}$$

where $L$ is the set of language pairs including inversions (i.e. six pairs for the three languages in Sami-Full), and $C_{AB}$ is the set of words in language $A$ that have cognates in language $B$. That is, for each word $c \in C_{AB}$ in a vocabulary $A$ and words $b$ in vocabulary $B$, we obtain the top $k$ ranking words $b \in B$, count how many of them are cognates to $c$ and divide this value by the total amount of cognates for $c$ in $B$. (For most words $c$, there is only one, and for some there are several cognates in $B$.) We then take the average of this value over $C_{AB}$, compute the corresponding value for other language pairs, and again take the average over language pairs.

### 4.3.4 Results

Having trained the S-CNN and SVM models on IE-Train, we run them and our baseline LDsim on the Sami language pairs in dataset Sami-Full and compute the MAR@$k$ values for $k = 1, \ldots, 100$. The resulting curves are shown in Figure 4.2.

We observe that all of the curves are logarithmic in shape. The S-CNN scores highest for all values of $k$, while LDsim scores lowest, and the SVM is approximately in the middle between the other two methods. The S-CNN and SVM curves rise quite steeply for $k \leq 20$, whereafter the rise is steady until $k = 100$. For LDsim, the rise is steeper only for very low values of $k$.

A steep rise in the MAR@$k$ curve means that when $k$ (i.e. the size of the retrieval set) increases, the occurrence of cognates within the first $k$ predictions increases more significantly. The rise is naturally steeper for small values of $k$, as $k$ increases faster in proportion to its value. With the steepest initial rise, S-CNN is most effective at ranking the correct cognates within the first $k$ candidates.

Since the S-CNN outperforms the other approaches by a substantial margin across values of $k$, it seems that the neural network is able to capture aspects of the cognacy relation that transfer across language families more effectively than the hand-designed features of SVM or the similarity metric LDsim. SVM also outperforms LDsim, which is unsurprising, since LDsim is included among its features. Since the S-CNN performs best at cognate identification when trained only on Indo-European data, we use it in our second experiment, where we fine-tune the model on a small set of Sami cognates.

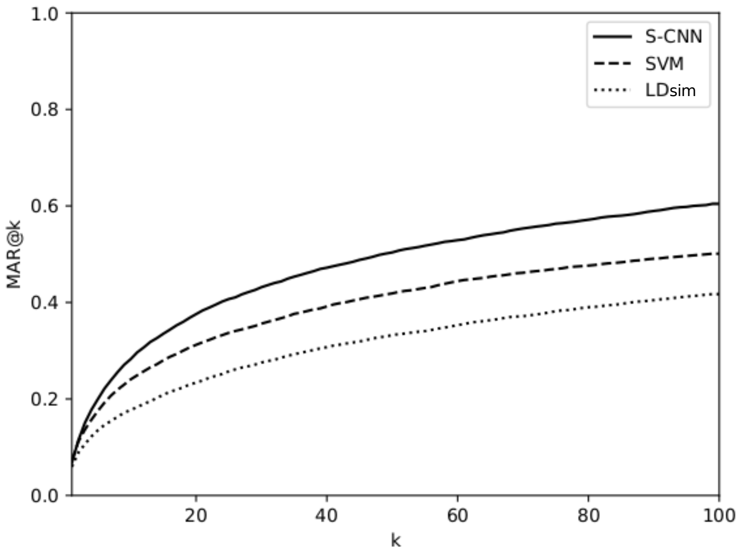

Figure 4.2: MAR@$k$ for $k = 1 \ldots 100$, for SAMI-FULL, for S-CNN and SVM trained on IE-TRAIN. (The baseline, LDSIM, requires no training.) Each curve is the average over all pairs of Sami languages.

## 4.4 Fine-Tuning on Sami Languages

In our first experiment in Section 4.3, our Siamese neural network model (S-CNN) performed best at direct transfer from Indo-European to Sami languages. Next, in our second experiment, we fine-tune S-CNN on a small set of Sami cognates in order to quantify the value of some labelled target-language data. We refer to this fine-tuned model as S-CNN-FT. We also analyse how the performance of S-CNN-FT improves with the amount of cognate pairs used for fine-tuning. In the following subsections, we describe the datasets, implementation, evaluation, and results of this experiment.

### 4.4.1 Datasets

We construct small-scale training and test sets from SAMI-FULL using the etymological database Álgu to include known cognate pairs in the sets. Table 4.2 shows the amounts of cognate and total word pairs in the fine-tuning and test sets, which we refer to as SAMI-FT and SAMI-FT-TEST. In these datasets, the proportion of cognate pairs is approximately 1%, which we consider suitable for the same reasons as in sampling IE-TRAIN, explained earlier in Section 4.3.1.

| **Dataset** | # cognate | # all pairs |
|---|---|---|
| Sami-FT | 986 | 100,000 |
| Sami-FT-Test | 3,500 | 350,000 |

Table 4.2: The small-scale datasets sampled from Sami-Full described in Section 4.3.1. We use Sami-FT to fine-tune the S-CNN pre-trained in the first experiment and test the fine-tuned model on Sami-FT-Test.

We generate negative (non-cognate) samples for both sets by taking random pairs from each language pair in Sami-Full, and checking that none of them is a known cognate pair. As explained in Section 4.3.1, due to the incompleteness of Álgu we cannot know with certainty that none of these random sample pairs would be cognate. However, for this experiment we decide to generate them in this way, since the vast majority of such pairs can be expected to be unrelated. That is, the results should still reliably indicate whether fine-tuning can improve performance and to what extent.

### 4.4.2 Implementation

When fine-tuning S-CNN pre-trained on IE-Train, the model's architecture is the same as described in Section 4.3.2. We do not freeze any layers for fine-tuning, but allow all weights to be updated. The hyperparameters we keep the same except for batch size (32) and number of epochs (20). The other methods (untuned S-CNN, SVM, LDsim) are the same as in the first experiment. The untuned S-CNN and SVM have been trained only on IE-Train.

### 4.4.3 Evaluation

In this second experiment, as mentioned in Section 4.4.1, we assume that only the known cognate pairs are truly cognates, and all other pairs are negative examples. This assumption allows us to use precision as an evaluation metric, which we posit is indicative of the impact of fine-tuning and tells us something additional about the methods' performance.

In order to evaluate our models at ranking the word pairs of Sami-FT-Test according to cognacy, we compute a precision-recall (PR) curve for each method. We consider this a suitable evaluation method as the classes are very imbalanced, and PR curves can indicate which methods are better than others at giving higher predictions to known Sami cognate pairs. The PR curve is created by calculating precision and recall at different proba-

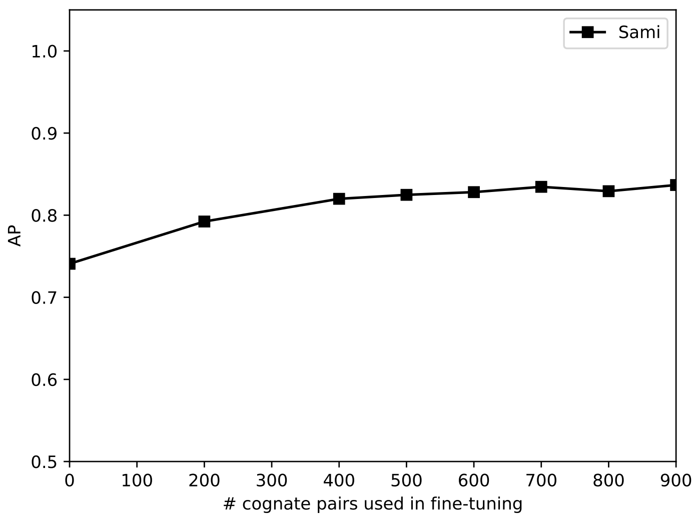

Figure 4.3: The learning curve of S-CNN fine-tuning on SAMI-FT in terms of average precision.

bility thresholds for predicting the positive class. Given a set of predictions in descending order of probability, we do this by computing the cumulative number of true and false positives for each threshold. (There are as many thresholds as there are unique predictions.) We summarise these curves with single values using *average precision* (AP) such that

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n$$

where $R_n$ and $P_n$ are the recall and precision at threshold $n$. That is, AP is the weighted mean of the precisions computed at each threshold, where the increases in recall are the weights. It is also equal to the area under the PR curve.

In addition to PR curves, we also compute the learning curve for the fine-tuned model (S-CNN-FT), by plotting the AP values for different amounts of cognate pairs from SAMI-FT used for fine-tuning. Thus, we can estimate how much fine-tuning data S-CNN-FT requires to achieve the performance illustrated by the model's PR curve.

### 4.4.4 Results

Figure 4.3 shows the learning curve of S-CNN-FT under fine-tuning. We observe that AP values increase together with the number of Sami cognates

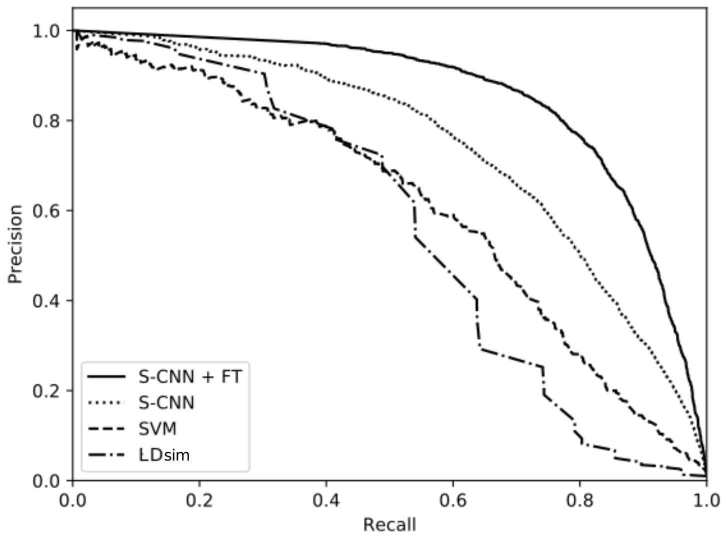

Figure 4.4: The PR curves for methods tested on Sami-FT-Test. All models are pre-trained on IE-Train, and S-CNN-FT is further fine-tuned on Sami-FT. (LDsim baseline requires no training.) Average precisions: **82.5%** (S-CNN-FT), 74.1% (S-CNN), 60.8% (SVM), and 54.0% (LDsim).

used to fine-tune the model, as expected. The improvement converges with about 500 cognate pairs, which implies that only a few hundred cognate pairs can be enough for maximal performance given a model pre-trained on an unrelated language family. Of course, in extremely low-resource scenarios, the available labelled data is often scarcer. However, the learning curve indicates that even smaller amounts ($< 200$) can result in noticeable improvements.

The PR curves for S-CNN-FT, unadapted S-CNN and SVM, as well as the LDsim baseline, are shown in Figure 4.4. In this case, S-CNN-FT has been fine-tuned with 500 cognate pairs, where the fine-tuning converges, as shown by the learning curve. Looking at these curves, and considering their respective AP values, we see that S-CNN-FT (AP = 82.5%) outperforms the unadapted S-CNN (AP = 74.1%) with a substantial margin. Otherwise, the results reflect those of the first experiment: S-CNN outperforms SVM and LDsim with a clear margin, and SVM is better than LDsim. However, the difference between SVM (AP = 60.8%) and LDsim (54.0%) is smaller, and for lower levels of recall ($R < 0.4$), LDsim obtains higher precision than SVM. As the precision of LDsim decreases faster for higher levels of recall, its overall AP score is still lower.

## 4.5 Conclusion

In this chapter, we have addressed the first research task of this thesis, RT-I:

**RT-I.** *Given scarce language-specific labelled data, identify cognates in truly low-resource Sami languages.*

Our first low-resource scenario has consisted of having a small cognate set for endangered South, North, and Skolt Sami of the Uralic language family, as our labelled target dataset. Being insufficient for direct supervised training, we have proposed a transfer learning approach based on pre-training models on data from other languages. Lacking high-resource close relatives, we chose the highly unrelated Indo-European language family as our source language set.

For cognate identification, we have proposed two models of similarity learning: a support vector machine (SVM) with string-metric features, and a Siamese convolutional neural network (S-CNN), transferring both of these from Indo-European to Sami. In our first experiment, we trained both models on Indo-European etymological data only, and applied them directly to identify cognates within the Sami language set. Our results showed that S-CNN, reaching highest recall, outperforms both the SVM and our baseline, LDsim. This indicates that the S-CNN more effectively generalises across language families, suggesting that it can learn patterns of cognacy that carry over across these two families.

Following this result, in order to utilise the small cognate set to adapt the pre-trained S-CNN to Sami, our approach is to fine-tune the model on this data. Analysing the model's ability to adapt, we found that already with 200 cognate pairs, fine-tuning resulted in a noticeable performance improvement. The model's learning curve converged at approximately 500 fine-tuning pairs. Using this amount, we compared its performance at classifying Sami cognates with the unadapted S-CNN and SVM models, as well as the LDsim baseline. The fine-tuned model was superior to its unadapted variant, and substantially so in comparison with SVM and LDsim.

Although neural networks have been applied relatively little to cognate identification, our results imply that as in many other NLP contexts, they provide promising solutions to the task, and potentially for historical linguistics in general. Since the convolutional neural network seems to capture such sound correspondences (predictive of cognacy) that carry over from Indo-European to Uralic languages, the method could possibly be useful for identifying even more universal patterns, which might be of interest for linguists. Additional benefits include providing a fast method to generate initial hypotheses regarding language relatedness and phylogenetic trees.

In terms of future work, varying the sets of source and target languages used for pre-training and testing could be a way to investigate more universal patterns of cognacy. Another research avenue is to combine this model with linguistic expertise and rule-based methods, for example using phonetically encoded inputs to mitigate the disparity between orthographic alphabets, probably resulting in sub-optimal performance for our models. Another limitation is that with regard to fine-tuning, even as few as 200 positive samples might not be available for many truly low-resource languages. In Chapter 5, we address exactly this issue and examine unsupervised adaptation.

From the results of this chapter, we conclude that we have developed a method for cognate identification in a truly low-resource-language scenario, using transfer learning in combination with a neural similarity learning model. Our results support our hypothesis that neural models are suitable for low-level NLP tasks in truly low-resource languages, when leveraged with transfer learning. Since our proposed approach does not require language-specific expertise, it is more scalable to new low-resource-language scenarios. In addition, it provides potential benefits for historical linguistics. Given that cognate information is useful for cross-lingual transfer learning, our work in this chapter supports the general extension of basic NLP tools into truly low-resource languages, in accordance with the objective of this thesis.

# Chapter 5

# Unsupervised Adaptation to Uralic Languages

In this chapter, we address cognate identification in endangered Sami and Finnic languages of the Uralic language family. Although this task is essentially the same as in Chapter 4, we examine a scenario where no labelled target-language data is available for fine-tuning. That is, we address our second research task, RT-II:

> **RT-II.** *Given only unlabelled data, adapt a pre-trained cognate identification model to truly low-resource Uralic languages.*

To accomplish RT-II, we investigate *unsupervised* methods of both *domain adaptation* (Section 2.2.4) and *cross-lingual adaptation* (Section 2.2.5), not relying on any labelled target data. Unsupervised methods are of special interest to truly low-resource languages, for which labelled data might be extremely scarce. Since even unlabelled data might be lacking in such languages, we focus on methods that require only moderate amounts of such data.

In this chapter, we investigate two unsupervised representation-based approaches to adapt our neural cognate identification model, S-CNN, from Indo-European to Uralic data. Our aim is to make source and target data distributions more similar by changing the underlying data representations, at two different levels. Then, a model trained on source data can be directly applied to target data, without a drop in performance associated with direct transfer of an unadapted model.

First, we consider *domain-adversarial networks*, a method of unsupervised domain adaptation based on latent feature learning. (See background in Sections 2.2.4 and 2.3.3.) We attempt to make S-CNN's latent word pair representations more similar to each other across source and target languages, by making target representations indistinguishable from source representations for a domain discriminator.

Second, we examine pre-trained *cross-lingual symbol embeddings*. These embeddings map language-specific symbols into a common, lower-dimensional vector space. This creates a new input encoding for S-CNN, making the alphabets of source and target languages more comparable for the model, especially if orthographies are different and alphabets do not overlap. Although embeddings of words are more prominent in NLP, we consider symbol embeddings more appropriate for our truly low-resource scenario. This is due to cognacy being determined by symbol and substring correspondences (see Section 3.1 for background), smaller training data requirements, and suitability for morphologically-rich languages, such as our Uralic target languages.

## 5.1 Problem

In this chapter, our problem setting is essentially the same as in Chapter 4. The Indo-European and Uralic language families again represent our source and target domains $\mathcal{D}_S$ and $\mathcal{D}_T$. Likewise, we address the same task $\mathcal{T}$, cognate identification as binary classification of word pairs. This task is the same in both source and target settings, $\mathcal{T}_S = \mathcal{T}_T = \mathcal{T}$. (See Section 4.1 for detailed descriptions of these domains and tasks.)

As in Chapter 4, source and target languages and marginal data distributions are different, implying that we face disparate feature spaces, $\mathcal{X}_S \neq \mathcal{X}_T$, marginal data distributions, $P_S(X_S) \neq P_T(X_T)$, as well as conditional distributions, $P_S(X_S|Y_S) \neq P_T(X_T|Y_T)$. Thus, we observe transfer learning conditions 1, 2, and 5 (see Section 2.2.2), and our transfer learning problem is a combination of domain adaptation and cross-lingual adaptation, similarly to Chapter 4. However, the adaptation problem is now more challenging, due to our assumption that our target dataset $D_T$ is completely unlabelled, and thus our source and target datasets are now

$$\begin{cases} D_S = \{(x_i^S, y_i^S)\}_{i=1}^{N_{D_S}} \\ D_T = \{x_i^T\}_{i=1}^{N_{D_T}} \end{cases}$$

where $N_{D_S}$ and $N_{D_T}$ are the sizes of these datasets. Since no labelled target data is available, direct supervision is not possible. That is, instead of fine-tuning a pre-trained model with a small amount of labelled data from $D_T$, we will represent the data so that both the distributions $P_S(X_S)$ and $P_T(X_T)$ and feature spaces $\mathcal{X}_S$ and $\mathcal{X}_T$ become more similar to each other. Then, we can apply a model trained on $D_S$ to classification of data in $D_T$.

## 5.2 Adversarial Adaptation to Sami Languages

Our first approach to unsupervised adaptation is to use *discriminative adversarial networks*, a method of latent feature learning. With this method, we aim to adapt the word pair representations of our cognate identification model, S-CNN, by mapping source and target word pairs closer to each other in a common representation space. We should learn such mappings $G_S(x_S)$ and $G_T(x_T)$ that map input word pairs $x = (x_1, x_2)$ into $d$-dimensional *latent features* $z \in \mathbb{R}^d$, such that the feature distributions

$$P_S(z_S) = \{G_S(x_S) \mid x_S \sim P_S(X_S)\}$$

$$P_T(z_T) = \{G_T(x_T) \mid x_T \sim P_T(X_T)\}$$

would be similar to each other, that is $P_S(z_S) \approx P_T(z_T)$, despite that $P_S(X_S) \neq P_T(X_T)$. We consider the S-CNN's merged word pair representations the latent features $z$, and the mapping $G$ as the part of the network encoding input word pairs into these representations. Next, in addition to the method, we present the setup and results of our experiments with adapting S-CNN from Indo-European to Sami data.

### 5.2.1 Discriminative Adversarial Networks

Our unsupervised adaptation method based on discriminative adversarial networks is illustrated in Figure 5.1. The method involves four models: *source encoder* $E_S$, *target encoder* $E_T$, *classifier* $C$, and *discriminator* $D$. While $E_S$ and $E_T$ encode source and target word pairs respectively into representations, $C$ and $D$ classify these representations into binary classes: $C$ in terms of cognacy, and $D$ in terms of domain label (source or target). This method is similar to the one used by Tzeng et al. [142] for image classification.

Before training, we split S-CNN into two separate models: an encoder $E$ and a classifier $C$. The encoder's input consists of word pairs $(w_1, w_2)$, represented with matrices $(\mathbf{X}_1, \mathbf{X}_2)$ where $\mathbf{X}$ is a matrix with one-hot column vectors representing a word's characters. As shown in Figure 4.1, each matrix is fed into a convolutonal layer with ReLU activation and max-pooling, resulting in a vector representation $\mathbf{r}$. The pair $(\mathbf{r}_1, \mathbf{r}_2)$ is then merged into one $h$-dimensional latent representation $\mathbf{z} \in \mathbb{R}^h$ using a *bilinear*

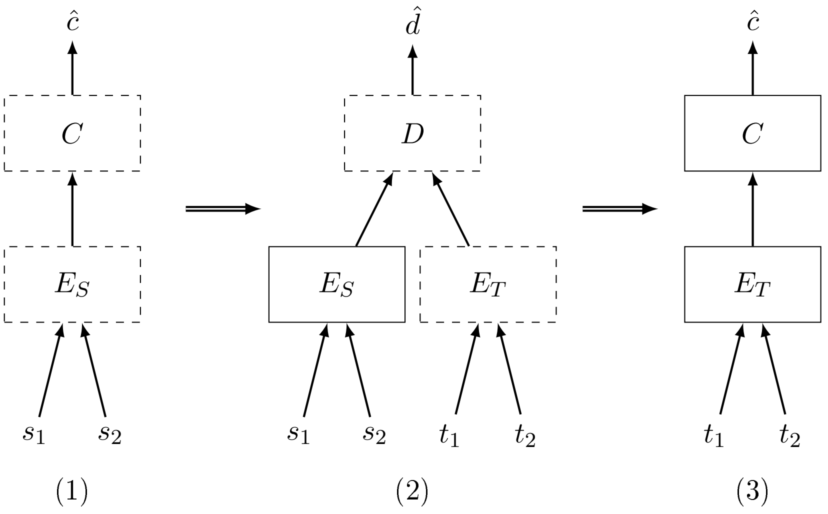

Figure 5.1: Our adversarial adaptation procedure consists of three stages: (1) pre-training source encoder $E_S$ and classifier $C$ with source word pairs $(s_1, s_2)$, (2) adversarial training of the target encoder $E_T$ and discriminator $D$, and (3) combining $E_T$ with $C$ in order to classify target domain word pair samples $(t_1, t_2)$. Dashed borders indicate models in training.

*transformation* such that

$$\mathbf{z} = \mathbf{r}_1 \mathbf{W}_m \mathbf{r}_2 + \mathbf{b}$$

where $\mathbf{W}_m \in \mathbb{R}^{h \times |\mathbf{r}| \times |\mathbf{r}|}$ is a weight tensor and $\mathbf{b}$ is the bias term of the merge layer. (We found a bilinear transformation resulted in better classification results than the absolute vector difference used in the previous chapter.) This latent word pair representation $\mathbf{z}$ is the output of the encoder and the input of the classifier and the discriminator.

Both the classifier $C$ and discriminator $D$ consist of fully-connected layers with ReLU activations. To the classifier's last fully-connected layer we also apply dropout as in the previous chapter. The output layer of both models contains two neurons for two classes, activated with the log-softmax function and producing predictions $y_i \in [0, 1]$ for each class $i$. While the classifier predicts whether two words are cognate or not, the discriminator predicts whether $\mathbf{z}$ is from source distribution $P_S(\mathbf{z_S})$ or target distribution $P_T(\mathbf{z_T})$ of the representation space.

Before adversarial training, we pre-train the source encoder $E_S$ and classifier $C$ together on our source domain dataset (IE-Train), in the same way we trained S-CNN in the previous chapter. Then, we initialise the target encoder $E_T$ with the weights of $E_S$. (The weights of $E_S$ are kept fixed during adversarial training, the second stage in Figure 5.1). We train the discriminator $D$ and $E_T$ adversarially so that while $D$ is learning to distinguish between the source and target representations $\mathbf{z}_S$ and $\mathbf{z}_T$, the weights of $E_T$ are updated towards the opposite direction. This makes it more difficult for the discriminator to classify target representations $\mathbf{z}_T$, and ideally they would shift closer to source representations $\mathbf{z}_S$.

To summarise, our adversarial training consists of the following steps:

(1) Obtain representations $\mathbf{z}_S$ and $\mathbf{z}_T$ from encoders $E_S$ and $E_T$,

(2) Compute the predictions $D(\mathbf{z}_S), D(\mathbf{z}_T)$ and update the weights of discriminator $D$,

(3) Invert domain labels of target representations $\mathbf{z}_T$, and

(4) Update the weights of encoder $E_T$ based on recomputed discriminator loss.

Finally, as illustrated by Figure 5.1, the adapted target encoder $E_T$ can be combined with the pre-trained classifier $C$ in order to classify target domain data.

### 5.2.2 Datasets

We train the source encoder $E_S$ and classifier $C$ on IE-Train. For testing, we construct a dataset of 100,000 word pairs, consisting of 3,986 known cognate pairs from Álgu (see Section 4.3.1), and randomly paired negative samples from Sami-Full. These randomly paired samples provide the data for adversarial training of target encoder $E_T$ and discriminator $D$. We sampled all word pairs from the three Sami languages (South Sami, North Sami, Skolt Sami) in proportion to their respective vocabulary sizes (see Table 4.1).

### 5.2.3 Implementation

Although we split the S-CNN in two parts, an encoder and a classifier, the architecture in both parts is the same as in Chapter 4, apart from the encoder's merge function. The encoder inputs are of the same form, as we continue to use one-hot-encoded character vectors. The height of an input matrix is $h = |\Sigma| = 336$, which is the size of the combined alphabet set of IE-Train and Sami-Full. Again, we fix the width to $n = 20$ characters, zero-padding shorter words while truncating longer ones.

In the convolutional layer, we use ten filters with height $p = |\Sigma|$ and width $q = 2$, and max-pooling with pool size $1 \times 2$. We set the dimension of the merged vector $\mathbf{m}$ (i.e. the size of the bilinear layer) to 100. In the classifier, we use one fully-connected layer with 100 neurons with a dropout rate of 0.5, while the discriminator has two fully-connected layers, also of size 100, without dropout.

We train the source encoder and the classifier on IE-TRAIN using cross-entropy loss for $C = 2$ classes and the Adam optimiser [71] with learning rate $\alpha = 0.0001$ and parameters $\beta_1 = 0.5$ and $\beta_2 = 0.9$. We set the batch size and number of epochs at 64 and 10 for training the source encoder and classifier, while we used batch size 8 and 1000 iterations for the adversarial training of the target encoder. We chose the training parameters for training the source encoder and classifier based on cross-validation with five folds, while for adaptation we used the same learning rate and optimiser parameters as Tzeng et al. [142]. We implemented the models with the PyTorch library [104].

### 5.2.4 Evaluation

We evaluate and compare two different encoder–classifier combinations: source and target encoders $E_S$ and $E_T$ combined with the same classifier $C$. As in the previous chapter, we use precision–recall curves in order to evaluate the models' performance.

### 5.2.5 Results

Figure 5.2 shows the precision–recall curves for the unadapted (pre-trained only) and adapted models. Combined with the classifier $C$ trained only on source data, the adapted target encoder $E_T$ achieves a clearly larger area under the curve than source encoder $E_S$. In terms of average precision (AP), the adaptation is able to improve the score from 58.8% to 70.8%. This suggests that discriminative adversarial networks are able to make the initially disparate Sami and Indo-European representations more similar to each other, resulting in an easier task for the pre-trained classifier $C$.

## 5.3 Cross-Lingual Adaptation to Finnic Languages

In this section, we present our second approach to unsupervised adaptation: pre-trained cross-lingual symbol embeddings. These embeddings provide a common representation space for the symbols observed in source and target datasets, making them comparable across languages and providing

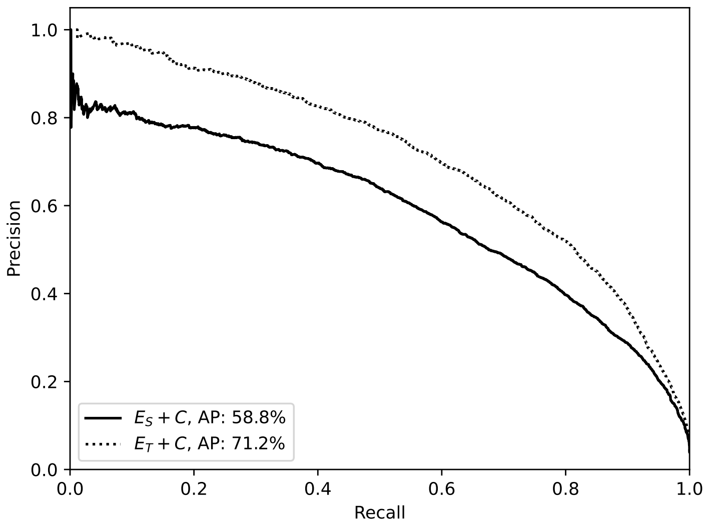

Figure 5.2: Precision–recall curves for unadapted source encoder $E_S$ and adapted target encoder $E_T$, both combined with the same classifier $C$.

a new input encoding for our cognate identification model, S-CNN. After describing these embeddings, we present our experiments with transferring S-CNN from Indo-European to a set of Finnic languages of the Uralic family.

### 5.3.1 Cross-Lingual Symbol Embeddings

Based on the principle that words occuring in similar contexts are more likely to have similar meanings, word embeddings have been very successful at capturing semantic relationships between words. However, cognacy is determined by cross-lingual relationships between sub-words and individual characters, as explained earlier in Section 3.1.

Granroth-Wilding and Toivonen [44] found that, in the same way as word embeddings capture word meaning based on co-occurrence, *symbol embeddings* can capture meaningful co-occurrence of characters and sub-words within a language. In addition, mapped into a common vector space, cross-lingual symbol embeddings for a language pair may indicate characters that are similarly used within both languages. Such cross-lingual character pairs would probably mark regular sound correspondences, indicative of etymological relatedness (see Section 3.1 for background).

As our second approach to unsupervised adaptation from Indo-European to Uralic languages, we propose to replace the one-hot-encoded inputs of

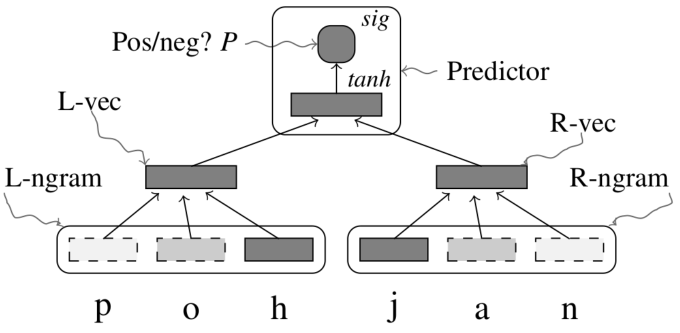

Figure 5.3: The XSYM model [44]. In this example, the word *pohjan* is separated into two trigrams (L-ngram, R-ngram) fed to the left and right parts of the network. Through fully-connected layers, the trigrams are composed into representations *L-vec* and *R-vec*, the concatenations of which is predicted as either a real or a fake language sequence.

the S-CNN model with dense cross-lingual symbol embeddings trained with XSYM, an unsupervised representation learning method [44]. As illustrated by Figure 5.3, XSYM is a feedforward neural network, which takes a short sequence of characters as its input, composes the left and right parts of the sequence into representations, concatenates these, and predicts whether or not the sequence is real language or not.

XSYM is trained alternately with text from two corpora in different languages, using a sliding window over the text. For each text sample, a corresponding negative sample is a random text sequence (fake language). In addition to individual characters, the input sequence length is varied to include also character bi-grams and tri-grams, as in Figure 5.3. All characters and $n$-grams are specific to one language, that is, the letter *a* of Finnish is considered distinct from the letter *a* of Estonian. Although the neural network's input consists of only one language at a time, by limiting the neural network's capacity, the model is forced to share information between the languages at the embedings level. This enables learning cross-lingual relationships between characters in an unsupervised way. We use the resulting embeddings directly as inputs to our S-CNN cognate classification model. We refer the reader to the paper of Granroth-Wilding and Toivonen [44] for further details about the XSYM model.

| | **Training** | **Testing** |
|---|---|---|
| **Dataset** | #pairs (#cognates) | #pairs (#cognates) |
| ES–PT | 326,040 (32,641) | 81,510 (8,114) |
| DA–SV | 78,264 (7,896) | 19,566 (1,887) |
| FI–ET | 2,240 (229) | 560 (51) |
| IZH–FI | 1,888 (183) | 472 (53) |
| IZH–KRL | 1,827 (177) | 203 (26) |

Table 5.1: Sizes of the different language-pair training and test sets, constructed using CogNet [9] and Álgu [61].

### 5.3.2 Datasets

Our experiments include five language pairs. Two of these are from the Indo-European family, namely Spanish–Portuguese (ES–PT) and Danish–Swedish (DA–SV). The other three are from the Finnic family, including Finnish–Estonian (FI–ET), Ingrian–North Karelian (IZH–KRL), and Ingrian–Finnish (IZH–FI). While Ingrian and North Karelian are truly low-resource, Finnish and Estonian are relatively high-resource languages, but their online etymological resources are relatively scarce for our purposes. For these five language pairs, pre-trained symbol embeddings are freely available[1], and cognate sets for them were found in CogNet [9] and Álgu [61]. The sizes of our datasets are shown in Table 5.1.

#### Pre-Trained Symbol Embeddings

The XSYM embeddings have been trained on the following corpus pairs: Spanish Europarl–Portuguese Europarl (ES–PT), Danish Wikipedia–Swedish Europarl (DA–SV), Ingrian Bible–Ylilauta forum (IZH–FI), Ingrian Bible–Olonets Karelian Bible (IZH–KRL), and Finnish Ylilauta forum–Estonian Reference Corpus (FI–ET). (We use Olonets Karelian embeddings for North Karelian, as these languages, or dialects, are very closely related and use the same alphabet. CogNet also did not contain any cognates for Ingrian–Olonets.) These embeddings are for individual characters only.

#### Training and Test Sets

To build our training and test sets, we retrieved the Indo-European cognates (ES–PT, DA–SV) from CogNet, and the Uralic cognates (FI–ET, IZH–FI,

[1] At `https://mark.granroth-wilding.co.uk/papers/unsup_symbol/`

izh–krl) from both CogNet and Álgu. For each dataset, we generate negative samples by pairing unrelated words randomly, so that approximately 10% of the word pairs are cognates. In Table 5.1, we report the training and test set sizes for each language pair.

CogNet, in the same way as the Etymological WordNet [39] used in our previous experiments, is an automatically built large-scale database of cognates where cognate pair has a common meaning. In contrast to Etymological WordNet, CogNet involves several sources of evidence in addition to Wiktionary: orthographic, semantic, and geographic evidence. In this work we use the most recent and largest version of CogNet (v2)[2]. The accuracy of the database is 95%, meaning that 5% of its 'cognates' are in fact not cognates. However, we opt to use this database in this chapter, as it is much larger than and overlapping with Etymological Wordnet, allowing us to sample larger datasets for specific language pairs instead of using all language pairs as we did with IE-Train in Chapter 4. Our other data source, Álgu, was described in Section 4.3.1.

### 5.3.3 Implementation

In this experiment, we use the same S-CNN model as in Section 5.2, although we do not separate the model into an encoder and a classifier. In other words, the model is the same as in Chapter 4, but with a bilinear layer as the merge function, instead of absolute vector difference. We keep the model's architecture the same, except for the height of the convolution kernel, $p$, which we adjust to match the input embedding dimension. For one-hot encodings, this is equal to the total alphabet size of training and test sets ($p = |\Sigma|$), while for Xsym embeddings this is $p = 30$. As before, the maximum word length is 20 characters, as almost all words in our datasets are shorter than this.

In order to use Xsym embeddings together with S-CNN, we encode each character found in a dataset with a pre-trained embedding, and encode input words by concatenating their characters' embeddings. In case there is no embedding for a certain character, we use a random vector with the same dimension.

When training the models, we varied the batch sizes and number of epochs according to dataset size. For the larger Indo-European sets (es–pt, da–sv) we set batch size to 64 and number of epochs to 20, while for the smaller Uralic sets (fi–et, izh–fi, izh–krl) we used batch size 4 and 30 epochs. We chose these values based on cross-validation on our training

[2]Available at: `http://ukc.disi.unitn.it/index.php/cognet/`

sets. Otherwise, as in Chapter 4, we use binary cross-entropy loss and the Adadelta [157] optimiser with $\alpha = 1.0$ and $\rho = 0.95$, and implemented the model with the PyTorch library [104].

### 5.3.4 Evaluation

Similarly to Chapter 4 and Section 5.2, we test our models on transfer across language families. However, we do this both ways between Indo-European and Uralic families. We also run some additional tests both within individual language pairs as well as across different language pairs within the same family, in order to test how performance is affected by language distance and the extent of data distribution mismatch. In contrast to previous cognate identification experiments, we do not use IE-TRAIN as our Indo-European dataset, since that would require pre-trained symbol embeddings for all language pairs in the dataset. Instead, our Indo-European set consists of two language pairs (ES–PT, DA–SV) and our Finnic set of three pairs (FI–ET, IZH–FI, IZH–KRL).

In this experiment, we compare the performance of three models, all being variants of S-CNN but using different ways of encoding input words. The first model (EMB) encodes inputs with pre-trained XSYM embeddings. The two other models, as previously, use binary one-hot vectors, whose dimension is equal to the total amount of different characters appearing in a dataset, and the vector's entries are $x_i = 1$ at one character-specific index and zeros otherwise ($x_j = 0, j \neq i$). The difference between these two models is that one (ONE-HOT) considers identical characters to be the same across languages (e.g. the Finnish and Estonian letter *a* is considered the same, i.e. $a_{fi} = a_{et} = a$), while the other (ONE-HOT-D) considers every character to be specific to a language (i.e. $a_{fi} \neq a_{et}$), in the same way as in XSYM embeddings.

We evaluate these models on three kinds of test setups: (1) within individual language pairs, (2) across language pairs within one family, and (3) across language families. For each test setup we compute the $F_1$ score as well as average precision (AP). We consider these metrics suitable in this experiment, as classes are imbalanced in the datasets. The $F_1$ score summarises the models' precision and recall for the test sets, while AP indicates how well they are able to rank word pairs according to cognacy, as explained in the previous chapter.

### 5.3.5 Results

Table 5.2 shows the $F_1$ and $AP$ scores for different S-CNN variants and test setups. We find that in general, ONE-HOT quite clearly outperforms the other two variants, while EMB is superior to ONE-HOT-D.

In the first setup, where train and test sets come from the same language pair, performance is highest for all three variants. Here, in terms of AP, the above pattern (ONE-HOT $>$ EMB $>$ ONE-HOT-D) is present in all language pairs except in IZH–FI, where EMB has the lowest value. Even then, its $F_1$ score is higher than that of ONE-HOT-D.

In the second setup, the variants are trained and tested on a different pair within the same language family. For the Indo-European pairs, the general level of performance drops significantly for EMB and ONE-HOT-D, while that of ONE-HOT only moderately. Here, EMB reaches a substantially higher $F_1$ and AP scores than ONE-HOT-D for both ways of transfer between ES–PT and DA–SV. However, when testing on Finnic pairs, the performance of EMB and ONE-HOT-D falls even further. Now EMB is only slightly better in terms of AP for IZH–KRL and FI–ET. Meanwhile, there is only a small difference for ONE-HOT except when tested on FI–ET.

In the third setup, the variants are trained and tested on different language families. In terms of AP values, the performance of EMB and ONE-HOT-D is similar when tested within the Uralic family, EMB reaching slightly higher scores except for $F_1$ when tested on Indo-European. The performance of ONE-HOT is now worst in terms of AP.

Altogether, the results in Table 5.2 show the extent to which performance drops along with increased distance between source and target languages. We consider it unsurprising that ONE-HOT-D performs worst across all setups, as it considers every character language-specific, assuming no cross-lingual character equivalences whatsoever. Regarding the significantly inferior performance of ONE-HOT-D, especially for cross-lingual tests, we note that the training sets might be too small for it, as its one-hot vectors are very high-dimensional, equal to the sum of language-specific alphabets.

We note that the superior performance of ONE-HOT probably depends on the fact that all five language pairs have similar Latin alphabets. This means that assuming symbol equivalence is a good strategy for our test sets, which is confirmed by our results. The result that EMB mostly outperforms ONE-HOT-D suggests that in cases where alphabets are more distinct (i.e. when orthographies are different), it would probably be the best-performing variant. That is, the more distinct the alphabets, the greater the similarity between ONE-HOT and ONE-HOT-D, while the performance of EMB should not change significantly.

| | | EMB | | ONE-HOT | | ONE-HOT-D | |
|---|---|---|---|---|---|---|---|
| | **Train–test pair** | $F_1$ | $AP$ | $F_1$ | $AP$ | $F_1$ | $AP$ |
| (1) | ES–PT → ES–PT | 90.1 | 95.9 | 91.1 | 96.2 | 89.1 | 94.0 |
| | DA–SV → DA–SV | 90.1 | 95.2 | 90.7 | 95.8 | 87.6 | 93.3 |
| | FI–ET → FI–ET | 47.5 | 67.9 | 63.1 | 79.3 | 16.7 | 43.3 |
| | IZH–FI → IZH–FI | 53.7 | 52.8 | 53.0 | 67.7 | 49.2 | 70.7 |
| | IZH–KRL → IZH–KRL | 65.1 | 68.0 | 57.3 | 80.1 | 62.5 | 64.4 |
| (2) | ES–PT → DA–SV | 36.7 | 30.5 | 80.0 | 86.1 | 0 | 9.64 |
| | DA–SV → ES–PT | 42.4 | 42.12 | 81.2 | 89.0 | 0 | 10.6 |
| (2) | FI–ET+IZH–FI → IZH–KRL | 2.7 | 11.1 | 75.7 | 84.5 | 14.9 | 9.4 |
| | FI–ET+IZH–KRL → IZH–FI | 5.9 | 9.7 | 77.8 | 86.9 | 6.8 | 11.5 |
| | IZH–FI+IZH–KRL → FI–ET | 8.2 | 11.7 | 54.0 | 60.9 | 0 | 8.9 |
| (3) | IE → Uralic | 11.2 | 11.8 | 67.5 | 74.7 | 0 | 10.0 |
| | Uralic → IE | 11.5 | 13.5 | 65.2 | 71.2 | 17.6 | 9.9 |

Table 5.2: $F_1$ and average precision scores in different transfer setups for S-CNN variants using three different input word representations: XSYM embeddings (EMB), one-hot vectors with symbol equivalence assumption (ONE-HOT), and one-hot vectors, all symbols distinct (ONE-HOT-D). Setups: (1) Training and testing within the same language pair, (2) training on one or several language pairs and testing on another from the same family, and (3) training on all pairs of one family, testing on the other family.

## 5.4 Conclusion

In this chapter, we have addressed our second research task, RT-II:

> **RT-II.** *Given only unlabelled data, adapt a pre-trained cognate identification model to truly low-resource Uralic languages.*

In this low-resource scenario, we have continued with the same task as in Chapter 4, cognate identification between truly low-resource Uralic languages. However, instead of having access to small cognate sets to use for adaptation (e.g. fine-tuning), we address an even more extreme scenario where only unlabelled data in moderate amounts is available for our target languages. Therefore, our transfer learning approach has been to use unsupervised methods of adaptation.

We have proposed discriminative adversarial networks and pre-trained cross-lingual symbol embeddings. We have conducted two experiments, in both of which we use the same similarity learning model we found to perform well at cognate identification in Chapter 4, the Siamese convolutional neural network (S-CNN).

In our first experiment, the adaptation was done at the level of merged word pair representations, the latent features. Initially, we pre-train S-CNN on Indo-European etymological data, and split the model into a source encoder and a classifier. Then, we initialise a target encoder with the source encoder's weights, and train it adversarially with a discriminator on unlabelled word pairs between South, North, and Skolt Sami. Testing the target encoder combined with the pre-trained classifier, we observe a noticeable improvement at ranking Sami word pairs according to cognacy. This result suggests that adversarial adaptation is able to make Sami and Indo-European representations more similar to each other, resulting in an easier task for the pre-trained classifier.

In our other experiment, we compare two kinds of representations to use as the S-CNN's inputs: pre-trained symbol embeddings and one-hot vectors. We run a set of tests with different combinations of source and target language pairs, including high-resource Spanish–Portuguese and Danish–Swedish of the Indo-European family, as well as the Finnic pairs of Finnish-Estonian, Ingrian–Finnish, and Ingrian–Karelian. Overall, we find that using pre-trained symbol embeddings results in better ranking performance (AP) in comparison with one-hot vectors when assuming no cross-lingual symbol equivalence. However, for our test sets, assuming such equivalence does result in best performance when using one-hot vectors. This is probably due to highly overlapping alphabets across all five language pairs, and an apparently high correspondence between similar characters. However, it is likely that for languages with disjoint alphabets, embeddings would result in best performance.

Although our results are encouraging, further research is needed to investigate the extent to which our methods generalise across other languages and families, or larger sets of language pairs within the same families. In particular, the potential of cross-lingual symbol embeddings should be examined with languages using different orthographies. Furthermore, our two methods could be combined: pre-trained embeddings with adversarial networks, or adversarial training to adapt embeddings before using them for classification.

We conclude that we have accomplished RT-II by developing methods of unsupervised cross-lingual adaptation for cognate identification between truly low-resource languages, We have considered transfer across language pairs both within and between the Indo-European and Uralic language families. We consider our methods scalable to new language settings, as no linguistic expertise is required. They require relatively small amounts of unlabelled language-specific data, including corpora to pre-train character embeddings. Altogether, our work in this chapter contributes to our objective of extending the coverage of low-level NLP to truly low-resource languages.

# Chapter 6

# Neural Document Planning for News Generation

In this chapter, we examine the task of document planning in data-to-text natural language generation (NLG) of news. In Section 3.2, we presented the composition of an NLG pipeline, where document planning consists of content selection and document structuring, that is, determining what information is presented in an output and in which order. As stated in Chapter 1, we address our third research task RT-III:

**RT-III.** *Given only auxiliary data, develop a method for document planning in news generation.*

We investigate especially neural methods, as they provide a promising avenue for more domain-independent and robust solutions to document planning [116] than traditional hand-engineered approaches [137].

However, in our low-resource scenario, we have no annotated training data for direct supervised training. In fact, a significant obstacle to the use of end-to-end neural approaches to document planning is the lack of suitable training data. In our news context, this would be professionally written news reports aligned with the data records that led to the reports. The availability of such datasets is in general very limited [57, 137], except for certain domains [e.g. 114].

Our approach to accomplish RT-III is to investigate the use of *distant supervision*: given unlabelled auxiliary data in the form of a news corpus, we construct a labelled training dataset for the task of *sentence ordering*, our source task. Having trained a neural model for this task, we transfer it to *document planning* in news generation, our target task. As newsrooms have extensive archives of reports written by journalists, that is, the process outputs, we consider news generation an appropriate application for distant supervision. Thus, we provide more insight into the applicability of distant supervision in low-resource NLP, an open research question [55]. Since our method requires only an unlabelled corpus, it is relatively domain-independent and thus scalable to new scenarios with low-resource domains. Thereby we contribute to the objective of this thesis, to widen NLG coverage over low-resource domains where it is often lacking [57].

We proceed with a description of our problem setting and the proposed sentence ordering models. Then, we present two experiments: first, we train the models on automatically constructed datasets, and test them on our source task, sentence ordering, to find out which one of them is most suitable for our target task, document planning. Second, we apply the chosen model (a Siamese convolutional network) to the document planning stage of an existing news generation pipeline [84], with a human evaluation of the outputs.

## 6.1 Problem

Our objective in this chapter is to apply a neural approach to the document planning stage of the news generation pipeline of Leppänen and Toivonen [84]. As discussed in Section 3.2, document planning consists of content selection and document structuring, that is, to select and order the information included in an output. In this chapter, our task is to select and order *messages*, roughly corresponding to sentences (see Sections 3.2 and 6.4.1), by computing specific importance weights for each message. Our aim is to do this using deep learning, in a truly low-resource scenario without any annotated training data for the task. However, unlabelled auxiliary data exists in the form of a news corpus from a similar domain.

As discussed in Section 3.2.1, the pieces of information in a news article tend to be arranged according to their importance, reflecting newsworthiness [138, 148]. Thus, we consider the task of *sentence ordering* an appropriate auxiliary task for document planning. Although we acknowledge that importance does not always decrease linearly from one sentence to the next, we postulate that by learning patterns between sentence position and content, we can also learn about the relationship between sentence content and importance, given enough training data.

From the perspective of transfer learning (see Section 2.2), we consider sentence ordering our source task $\mathcal{T}_S = \{\mathcal{Y}_S, P_S(Y_S), P_S(Y_S|X_S)\}$, and document planning our target task $\mathcal{T}_T = \{\mathcal{Y}_T.P_T(Y_T), P_T(Y_T|X_T)\}$. These tasks differ in their label spaces, as the source task is to predict sentence position while target task is to predict the message weights. That is, source label space is of the form $\mathcal{Y}_S = \{0, \ldots, C-1\}$, where $C$ is the maximum number of sentences in an input paragraph. The target label space is $\mathcal{Y}_T = \mathbb{R}_+$, as the weights are positive. Clearly, $\mathcal{Y}_S \neq \mathcal{Y}_T$, from which follows that prior and conditional label distributions are also different, $P_S(Y_S) \neq P_T(Y_T)$ and $P_S(Y_S|X_S) \neq P_T(Y_T|\ X_T)$. That is, conditions 3–5 described in Section 2.2.2 are present.

Our auxiliary data is a news corpus consisting of news reports on various statistics about the state of Finland, as will be explained in Section 6.4. In contrast, the dataset underlying our news generation experiment concerns a more specific topic, consumer price indices of various product categories. Both the corpus and generation are in English, implying that source and target feature spaces are the same, $\mathcal{X}_S = \mathcal{X}_T$. Also, the data distributions $P_S(X_S)$ and $P_T(X_T)$ are similar, although generation is concerned with only a subset of the corpus's topics.

Hence, in this low-resource scenario, we face the condition of disparate target tasks, addressed by *inductive* transfer learning (see Section 2.2.3). However, in contrast to previous work assuming the availability of labelled data for the target task [123], we only have unlabelled auxiliary data.

Our approach is to use *sequential* transfer learning with *distantly supervised* pre-training: we automatically annotate a labelled source dataset $D_S$, and train a neural model for sentence ordering, $\mathcal{T}_S$. We construct $D_S$ separately for each model variant presented in the next section. Although the generation database consists of numerical data, the generation pipeline converts numerical data points to messages in sentence format. We consider this resulting dataset our target dataset $D_T$. In our news generation experiment (Section 6.4), we develop a scheme to weight the messages in $D_T$ using a neural sentence ordering model, thus transferring it to our target task $\mathcal{T}_T$, document planning.

## 6.2 Neural Networks for Sentence Ordering

In this section, we present three different neural network models we train for our source task $\mathcal{T}_S$, sentence ordering. In our first experiment in Section 6.3, we use these models to order sentences within paragraphs of raw human-written statistical news articles. Common to these models is that they are all trained to predict the 'correct' position of an input sentence within a paragraph, that is, the sentence's most appropriate position given its content.

Each of the following neural network models has a different approach to the sentence ordering task. The first one, a *Siamese* network, resembling the one we applied to cognate identification in Chapters 4–5, is a binary classifier of sentence pairs predicting whether one sentence should precede another. The second one, a *positional* network, is a multi-class classifier predicting the most likely paragraph quantile to which a sentence should belong. The third one, a *pointer* network, is a sequence-to-sequence model predicting the correct ordering of a shuffled paragraph of sentences. While the pointer network can be directly applied to sorting a paragraph, the first two models require an additional sorting operation in order to perform sentence ordering.

We examine these three models, as they represent different lines of recent work where similar models have been proposed (see background in Section 3.2). Since our end goal is to apply one of these models to document planning for news generation (Section 6.4), we are more interested in how these models compare with each other, than in reaching state-of-the-art performance for sentence ordering.

### 6.2.1 Siamese Network

The Siamese neural network is designed to learn the similarity between to inputs, as discussed in Section 2.1.4. In Chapters 4–5, we predicted the cognacy of word pairs with such a model. In this chapter, we investigate whether it can learn which one of two sentences should precede the other. Given such pairwise comparisons, the model can be used to sort a set of sentences.

The model takes a sentence pair $(s_i, s_j)$ as input, and outputs a value $\hat{y} \in [0, 1]$, its prediction of whether $s_i$ should precede $s_j$ in a paragraph. Figure 6.1 illustrates a sentence pair $(s_1, s_2)$ being first encoded into matrices $(\mathbf{X}_i, \mathbf{X}_j) = ([\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in}], [\mathbf{x}_{j1}, \ldots, \mathbf{x}_{jm}])$ representing the sentences as token embedding sequences of lengths $n$ and $m$. Each sequence is then transformed into one representation of the whole sentence, using either a convolutional or a BiLSTM layer, resulting in a pair of representation vectors $(\mathbf{s}_i, \mathbf{s}_j)$. This vector pair is then further merged into one vector $\mathbf{m} \in \mathbb{R}^h$ through a bilinear transformation such that $\mathbf{m} = \mathbf{s}_i \mathbf{W}_m \mathbf{s}_j$, where $\mathbf{W}_m \in \mathbb{R}^{h \times n \times n}$ is a weight tensor and $\mathbf{s}_i, \mathbf{s}_j \in \mathbb{R}^n$. Finally, this merged vector $\mathbf{s}$ is fed to a fully-connected layer, whereafter the final prediction $\hat{y}$ is computed using a sigmoid activation function.

We propose two variants of this Siamese network: convolutional (S-CNN) and recurrent (S-BiLSTM), each computing the sentence representation vectors $(\mathbf{s}_i, \mathbf{s}_j)$ differently. S-CNN does this with a convolutional layer similarly to our earlier cognate identification model convolving character vectors, by applying a set of $k$ kernels to each input of $(\mathbf{X}_i, \mathbf{X}_j)$. The resulting $k$ feature

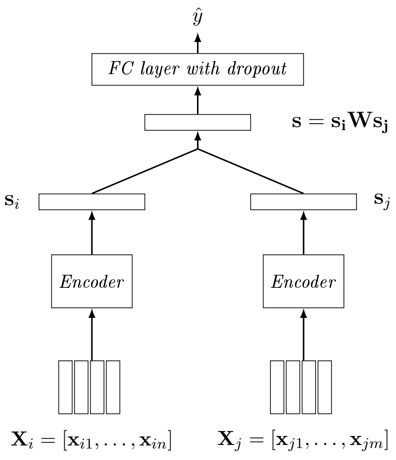
$\hat{y}$
*FC layer with dropout*
$\mathbf{s} = \mathbf{s_i W s_j}$
$\mathbf{s}_i$
$\mathbf{s}_j$
*Encoder*
*Encoder*
$\mathbf{X}_i = [\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in}]$
$\mathbf{X}_j = [\mathbf{x}_{j1}, \ldots, \mathbf{x}_{jm}]$

Figure 6.1: The Siamese network for sentence ordering. $\mathbf{X}_1$ and $\mathbf{X}_2$ denote the inputs of concatenated token embeddings, while $\mathbf{s}_1$ and $\mathbf{s}_2$ denote the representations resulting from either a convolutional or recurrent encoder layer, and $\mathbf{s}$ is the product of the bilinear operation between $\mathbf{s}_1$ and $\mathbf{s}_2$. Finally, $\hat{y} \in [0, 1]$ is the final prediction whether $s_1$ should appear before $s_2$ in a news article.

maps are concatenated into single vectors $(\mathbf{s}_i, \mathbf{s}_j)$. Meanwhile, S-BiLSTM uses a bidirectional LSTM network to produce sentence representations $(\mathbf{s}_i, \mathbf{s}_j)$ as concatenations of the last hidden states of both directions. While our two variants of the Siamese network are similar to earlier work [2, 24] (see background in Section 3.2.3), we propose modifications with a bidirectional LSTM and a bilinear merging of sentence representations.

To order a set of sentences with a Siamese network, we score each candidate next sentence of the correct ordering $O$ by comparing it with all other sentences within a paragraph, and taking the sum of the log-probabilities of all comparisons, similar to Chen et al. [24]. Thus we can sort the set according to these sums, which reflect the model's predictions of each sentence's correct position in the paragraph.

### 6.2.2 Positional Network

Our second model is a neural network that works as a sentence position classifier, following Bohn et al. [14], illustrated in Figure 6.2. Instead of sentence pairs $(s_i, s_j)$, it takes only one sentence at a time as input, and outputs a prediction $\hat{y} \in \{1, 2, \ldots, Q\}$, where $Q$ is the number of *quantiles* into which a paragraph of news text is divided. That is, this model does

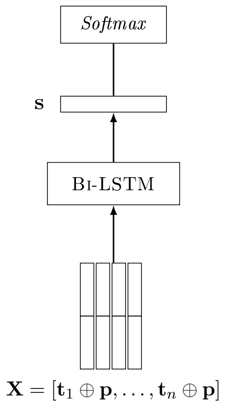
Softmax
s
Bi-LSTM
$\mathbf{X} = [\mathbf{t}_1 \oplus \mathbf{p}, \ldots, \mathbf{t}_n \oplus \mathbf{p}]$

Figure 6.2: The architecture of the positional neural network.

not predict the exact position of a sentence within a paragraph, but the most appropriate quantile for it. The number of quantiles has to be predetermined, and is fixed for all paragraphs in both training and testing phases regardless of the actual number of sentences in paragraphs.

As in the Siamese network, an input sentence $s$ is first encoded into a sequence of token embeddings $[\mathbf{t}_1, \ldots, \mathbf{t}_n]$. In addition, in this architecture each token embedding is concatenated with an embedding $\mathbf{p}$ representing the whole paragraph $p$ to which sentence $s$ belongs. This paragraph embedding $\mathbf{p}$ is the average of all tokens in the paragraph, $\mathbf{p} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{t}_i$. Thus, the input becomes $\mathbf{X} = [\mathbf{t}_1 \oplus \mathbf{p}, \ldots, \mathbf{t}_n \oplus \mathbf{p}]$. In this way, the network can relate the given input sentence to its context, wheras the Siamese network does this by comparing one sentence to another.

Next, the embedding sequence $\mathbf{X}$ is taken as input by a bidirectional LSTM layer, from which the last hidden states of both directions are concatenated into representation $\mathbf{s}$ of the sentence. As in the Siamese network, $\mathbf{s}$ is then further passed on to a fully-connected layer with dropout, and finally a softmax output vector $\hat{\mathbf{q}} = [\hat{q}_1, \ldots, \hat{q}_Q] \in [0, 1]^Q$ indicates the sentence's predicted quantile $\hat{y}$ such that $\hat{y} = \arg\max_i \hat{q}_i$.

With the positional network, we score each sentence $s_i$ of some shuffled paragraph $S$ by computing the weighted average of the predicted quantile for each $s_i$, following Bohn et al. [14], such that

$$\bar{y}(s_i) = \sum_{i=1}^{Q} i\hat{q}_i$$

and then sort $S$ from lowest to highest scoring sentences.

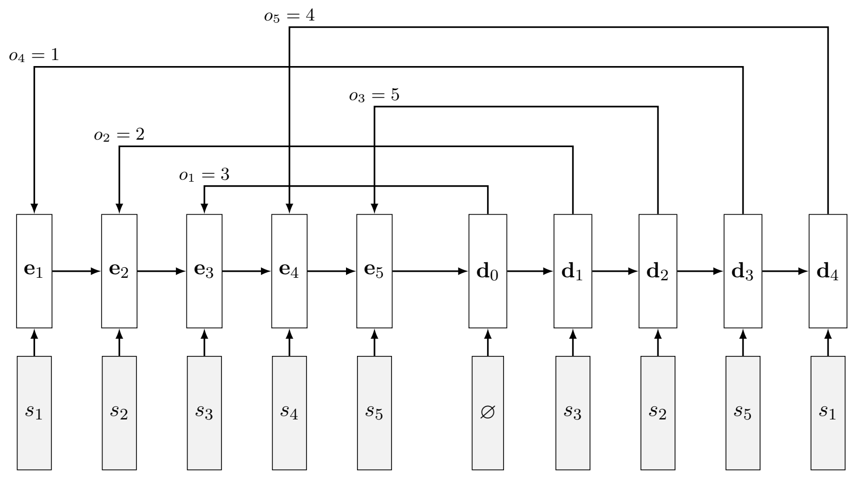

Figure 6.3: A pointer network [145] sorting sentences $S = s_1, \ldots, s_5$. At each decoding step $t$, attention is used to predict the index $o_t$ of the next sentence.

### 6.2.3 Pointer Network

Our third sentence ordering model is a sequence-to-sequence recurrent neural network and a variant of the *pointer network* first introducced by Vinyals et al. [145]. The purpose of this model is to predict the correct ordering $O = (o_1, \ldots, o_n)$ of an unordered set of $n$ sentences $S = \{s_1, \ldots, s_n\}$, such that the correct sentence sequence is $S' = (s_{o_1}, \ldots, s_{o_n})$. The architecture consists of an *encoder* and a *decoder*, which are both LSTM networks. The architecture is illustrated in Figure 6.3.

The encoder's input is a sentence embedding sequence $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_n)$ representing the unordered sentence set $S$, from which the encoder computes the corresponding hidden states $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_n]$, where the last hidden state $\mathbf{e}_n$ is also a representation of the whole input sequence. Given this last encoder hidden state $\mathbf{e}_n$, the decoder computes in $n$ steps its own hidden states $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_n]$ and outputs its prediction $\hat{O} = (\hat{o}_1, \ldots, \hat{o}_n)$ for the correct ordering of $S$.

At each decoding step $i = 1 \ldots n$, the decoder predicts the index $\hat{o}_i$ of the next sentence $s_{\hat{o}_i}$ in the correct ordering. The decoder's input consists of all the encoder hidden states $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_n]$, the decoder's hidden state $\mathbf{d}_{i-1}$ from the previous step, and the previously predicted sentence embedding $\mathbf{s}_{\hat{o}_{i-1}}$. The previous index $\hat{o}_{i-1}$ is also known, and cannot be predicted again. At the first step $i = 1$, the encoder's last hidden state is used as the decoder's initial hidden state ($\mathbf{d}_0 = \mathbf{e}_n$), and $\mathbf{s}_{\hat{o}_0}$ is a random vector.

At each step, the decoder uses an attention layer to compute a vector $\mathbf{u}_i$, attention vector $\mathbf{a}_i$, and a new hidden state vector $\mathbf{d}_i$ such that

$$\mathbf{u}_i = \mathbf{v}^T \tanh(\mathbf{W}_1 \mathbf{E} + \mathbf{W}_2 \mathbf{d}_{i-1})$$

$$\mathbf{a}_i = \mathrm{softmax}(\mathbf{u}_i)$$

$$\mathbf{d}_i = \sum_{j=1}^{n} a_{ij} \mathbf{e}_j \oplus \mathbf{s}_{i-1}$$

where $\mathbf{v}, \mathbf{W}_1, \mathbf{W}_2$ are the model's learnable parameters and $\oplus$ is a concatenation operation. The attention vector $\mathbf{a}_i$ can be regarded as a probability distribution over the input sentences, and the index of its maximum element gives the next sentence the network 'points' to. These indices are the predicted ordering $\hat{O} = (\hat{o}_1, \ldots, \hat{o}_n)$. Duplicate predictions are avoided by setting the probabilities of previously predicted indices to zero. The new hidden state $\mathbf{d}_i$ is a concatenation of the attention-weighted sum vector of the encoder's hidden states and the decoder input $\mathbf{s}_{\hat{o}_{i-1}}$.

## 6.3 Sentence Ordering Experiment

In this first experiment, we examine four neural models: Siamese BiLSTM (S-BiLSTM), Siamese CNN (S-CNN), positional network (Pos-Net), and pointer network (Pointer-Net). We test their performance at sentence ordering, our source task $\mathcal{T}_S$. Our aim with this experiment is to determine which one of these models we should use for our target task $\mathcal{T}_T$, document planning, in our news generation experiment in Section 6.4. Based on both performance and its otherwise advantageous architecture, we choose to use the Siamese neural network (S-CNN) in that experiment. Next, we describe our datasets, training implementation, evaluation, and results.

### 6.3.1 Datasets

For training, validation, and testing, we use a news text corpus consisting of web-crawled statistical news articles produced by Statistics Finland[1]. This corpus contains Finnish, Swedish, and English news articles, which report statistical news on various topics including economy, crime, accidents, and so on. The experiments presented in this chapter use only the English subcorpus. From now on, we refer to this corpus as StatFi.

We focus on ordering sentences within paragraphs, since our second experiment is concerned with the planning of paragraphs within news reports.

[1]`https://www.stat.fi/`

Nevertheless, we split StatFi by articles in order to prevent paragraphs from the same article (with similar content) ending up in both training and test sets. StatFi contains 4,383 news articles, of which we set aside 876 (20%) for testing, and use the remaining 3,507 for training and validation. The only pre-processing we do is filtering out sentences with fewer than three words, as well as paragraphs with one sentence only. This results in a total of 21,854 paragraphs (77,054 sentences), of which 4,567 paragraphs (16,028 sentences) for testing and 17,287 paragraphs (61,026 sentences) for training.

### 6.3.2 Implementation

#### Contructing training data

For each sentence ordering model, we automatically annotate a training dataset using StatFi in a different manner, aiming to provide a good learning signal with regard to sentence content and position.

As a pairwise model, S-BiLSTM and S-CNN (Section 6.2.1) are trained on sentence pairs. We sample the training set for this model by splitting paragraphs in half and forming sentence pairs $(s_i, s_j)$ where $s_i$ is from the first and $s_j$ from the other half. We label these pairs with $y = 1$ and the corresponding inversions $(s_j, s_i)$ with $y = 0$. (We also tried sampling sentence pairs from articles' first and last paragraphs, as well as between all sentence pairs within paragraphs, but found our choice give a better learning signal.)

For the positional neural network (Section 6.2.2), a multi-class classifier, we label each sentence $s_i$ of each paragraph $p$ with a quantile label $q_i$. To do this, we use the formula

$$q_i = \lfloor \frac{i-1}{\frac{n}{Q}} \rfloor \; \forall i = 1, \ldots, n$$

where $n$ is the number of sentences in a paragraph. That is, in case there are fewer sentences than quantiles, some quantiles will not be assigned to any sentence, while in the opposite case, several sentences might be assigned to the same quantile. For the pointer network, we shuffle the sentences of each paragraph and label the shuffled sentences with their indices in the correct ordering, that is, the original paragraph.

#### Sentence encoder

All of our neural architectures require that each sentence is encoded from string format into either one sentence embedding or a sequence of token embeddings. To do this, we tried out several pre-trained language models, namely BERT [32] and Elmo [108], as well as pre-trained word embeddings word2vec [97], GloVe [107], and fasttext [15]. Based on validation results, we chose to use Sentence-BERT [120] (S-BERT) for our reported experiments. S-BERT has been specifically designed for obtaining sentence embeddings and context-dependent token embeddings efficiently. In order to produce better embeddings, we fine-tune S-BERT on StatFi.

#### Hyperparameters

In the Siamese CNN, we use one convolutional layer with ten kernels, kernel size $1024 \times 6$ (input token embedding dimension is 1024), pooling size $1 \times 2$, and the length of the input sentence is limited to 30 tokens. (Shorter sentences are zero-padded, longer ones are truncated.) In the Siamese LSTM, we use a bidirectional layer where the hidden state vector has dimensionality of 80. Both networks have 64 hidden units in the fully-connected layer and the dropout parameter we set to 0.5. We choose these values based on validation results of sentence pair classification.

The positional network uses a bi-directional LSTM to compute a sentence representation, has a hidden state vector of dimension 100 and the number of quantiles we set to $Q = 10$. In the pointer network, the number of attention units is 100, hidden state vector size is 100 in the encoder and $100 + 1024$ in the decoder, after concatenation with a sentence embedding. The dimension of the input embeddings is 1024 in all cases. In the positional network, the input embedding is concatenated with that of the whole paragraph, and the dimension is doubled.

Training the models, we use binary cross-entropy loss function for the Siamese networks, and general cross entropy for both the positional and pointer networks. For model validation, we use cross validation by splitting the training set into five folds. In order to determine an appropriate number of training epochs, we use early stopping should the validation loss increase for two consecutive epochs. We train the final models using 10 epochs (20 for the pointer network) and batch size 32, Adadelta optimiser [157] with learning rate $\alpha = 1.0$ and parameters $\rho = 0.9$ and $\epsilon = 1e - 6$.

### 6.3.3 Evaluation

We test the performance of four models at the task of predicting the original ordering of sentences shuffled within a paragraph. As mentioned in Section 6.2, only the pointer network predicts an ordering directly, while the Siamese and positional networks orderings are based on log-probability scores and weighted average quantiles, respectively. We compare these models with random ordering.

We use the following evaluation metrics for the ordering task: Kendall's $\tau$, perfect match ratio (PMR), and positional accuracy (PAcc), which are computed as follows:

$$\tau = \frac{1}{|D|} \sum_{p \in D} \frac{2S(O_p, \hat{O}_p)}{\frac{|p|(|p|-1)}{2}} \in [-1, 1]$$

$$\mathrm{PMR} = \frac{1}{|D|} \sum_{p \in D} \mathbb{I}(O_p = \hat{O}_p)$$

$$\mathrm{PAcc} = \frac{1}{|D|} \sum_{p \in D} \sum_{i \in |p|} \mathbb{I}(o_{s_i} = \hat{o}_{s_i}),$$

where $D$ is the dataset of paragraphs, $p$ is a paragraph, and $S(O_p, \hat{O}_p)$ is the minimum number of adjacent transpositions required to change the predicted ordering $\hat{O}_p$ to the correct ordering $O_p$. $\mathbb{I}$ is the indicator function, equal to one when the condition is met and zero otherwise.

Kendall's $\tau$ measures rank correlation, and has been found to correlate with human judgements at information-ordering tasks [80]. PMR is the ratio of all correctly predicted orderings to all paragraphs in the test set, whereas PAcc is the ratio of correctly predicted sentence positions to all sentences in the test set.

### 6.3.4 Results

The results of the sentence ordering task are shown in Table 6.1. We observe that POINTER-NET reaches highest scores for all three metrics, meaning that its predictions correlate most with the original orderings, and have the highest ratios of both perfectly matching orderings and correctly positioned sentences. The S-CNN is second, reaching nearly the same scores. It is followed by S-BiLSTM and POS-NET, which have clearly lower scores, although clearly higher than random ordering in terms of Kendall's $\tau$.

These results indicate that the pointer network's sequence-to-sequence architecture combined with attention learns to make connections between sentences' position and content more effectively than the other models. This finding is in agreement with recent success on sentence ordering using pointer networks [42, 147], as discussed in Section 3.2.3. In addition, the good performance of S-CNN aligns with previous work on sentence classification with convolutional networks [69], and especially on pairwise modelling [151].

| **Model** | **Kendall's** $\tau$ | **PMR** (%) | **PAcc** (%) |
|---|---|---|---|
| Random | 0.006 | 23.4 | 30.45 |
| S-CNN | 0.414 | 42.0 | 44.6 |
| S-BiLSTM | 0.189 | 32.4 | 36.7 |
| Pos-Net | 0.128 | 28.9 | 34.6 |
| Pointer-Net | **0.438** | **44.0** | **45.9** |

Table 6.1: Results on the sentence ordering task for Siamese networks with CNN (S-CNN) and BiLSTM (S-BiLSTM) layers, positional network (Pos-Net), and the pointer network (Pointer-Net).

We speculate that these two models' higher performance at sentence ordering depends on their respective attention and convolutional layers. While they work differently, both mechanisms facilitate the recognition of whether certain features are present in the input, with less regard for their exact position. That is, the presence of certain words or phrases in a sentence are more easily learned by a convolutional kernel, and affect the sentence embeddings pointed to by Pointer-Net. Intuitively we also consider it likely that the presence of certain words or phrases is a more important predictor of sentence order and paragraph structure than their position within the sentence. This would also be in agreement with the findings of previous work [30, 102, 147].

Although Pointer-Net slightly outperforms S-CNN at sentence ordering, we consider S-CNN more suitable to our target task $\mathcal{T}_T$, document planning. With fewer parameters, S-CNN is computationally less complex. This makes it more scalable to other turly low-resource scenarios with even smaller amounts of auxiliary data, which is in accordance with the objective of this thesis. Furthermore, its lower complexity makes it more suitable for use cases of online generation upon a user's query, which is our setup in Section 6.4.

## 6.4 News Generation Experiment

In this section, we present our second experiment. Our aim is to evaluate a neural sentence ordering model, S-CNN, at our target task $\mathcal{T}_T$, document planning for news generation. Next, we describe the generation pipeline, the context of our expoeriment. Then, we present our method for document planning using S-CNN, as well as our human evaluation scheme and the results.

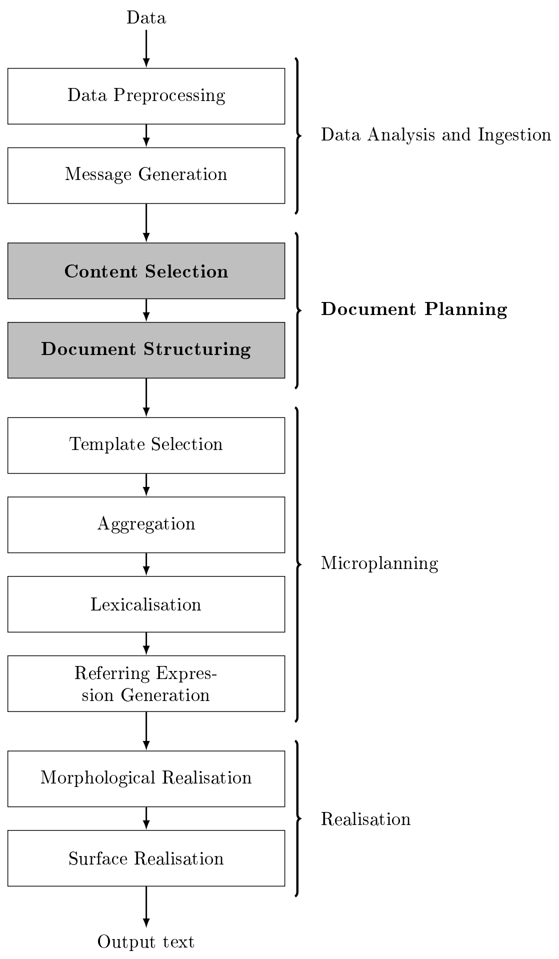

Figure 6.4: Overview of the NLG pipeline of Leppänen and Toivonen [84]. We modify only the document planning stage, using a neural sentence ordering model for this task.

### 6.4.1 Generation Pipeline

We apply our neural document planning approach to an adaptation of the tmeplate-based news generation system of Leppänen and Toivonen [84]. Figure 6.4 presents a high-level overview of this system. It is designed for data-to-text generation of short news reports based on structured statistical data, inlcuding four major stages: data analysis and ingestion, document planning, microplanning, and realisation. We provide a brief description of each stage, and for more details we refer the reader to their paper.

| Field | Description | Example value |
|---|---|---|
| `where` | What location the fact relates to | Finland |
| `where_type` | What the type of the location is | country |
| `timestamp` | The time (or time range) the fact relates to | 2020M05 |
| `timestamp_type` | The type of the timestamp | month |
| `value` | A numeric value | 0.01 |
| `value_type` | Interpretation of `value` | cphi:hicp2015:cp-hi02:rt01 |
| `newsworthiness` | An outlierness estimate | 1 |

Table 6.2: An example of a message stating that in the fifth month of 2020, in Finland, using the year 2015 as the start of the index, the consumer price index of alcoholic beverages and tobacco changed by 0.01 points with respect to the value of the index during the previous month. From the paper of Leppänen and Toivonen [84].

## Data Analysis and Ingestion

The data points underlying the generation are from the consumer price index database of the European statistics agency EuroStat, downloaded in September 2020[2]. It contains country-level data about the consumer price indices of various price categories for EU member countries and the US, and their monthly change over time starting from January 1996. We use the database as pre-processed by Leppänen and Toivonen [84], with the addition of monthly rankings of countries by price category (from greatest-to-lowest values) and comparisons to average values across the EU.

In data preprocessing, input data points are converted to a set of *messages*, data structures with semantic representations of facts treated as atomic units of information. An example message is illustrated in Table 6.2. Each message is associated with an *outlierness* value based on statistical outlierness with regard to the distribution of comparable messages, using the Interquartile Range (IQR) method [83]. These values are intended to reflect the messages' newsworthiness. These values are adjusted according to recency, considering more recent data points more newsworthy. Furthermore, each message is associated with a high-level *topic*, in our case corresponding to a specific consumer price index category (e.g. 'clothing and footwear'). In contrast to Leppänen and Toivonen [84], we consider only such high-level topics, instead of fine-grained label hierarchies. That is, we

[2]Available at: `https://appsso.eurostat.ec.europa.eu/nui/show.do?dataset=ei_cphi_m`.

do not assume the availability of the kind of detailed, structural metadata used in their work.

The generated messages are divided into two sets, the *core set* and the *expanded set*. The core messages pertain to a certain focus country, determined by user query, while expanded set contains messages corresponding to rest of the data in the database. For example, given 'France' as the user-selected focus country, the core set contains all messages about France, while the expanded set includes messages about all other countries. While the system should generate a text about a queried country, it is also intended to describe data concerning other countries, providing additional context. In our experiments, the core set tends to include several thousands of messages, while the expanded set is an order of magnitude larger. Although messages do not have any linguistic structure at this point, they can be conceptualised as phrases or short sentences.

### Document Planning

After data pre-processing and message generation, the core and expanded sets are processed into a *document plan*. This describes what information is to be included in the final output (content selection) and how it will be structured (document structuring). In this work, we apply our neural sentence ordering model to this document planning stage, recomputing the initial outlierness values reflecting messages' newsworthiness.

Given the two sets of messages, the first step is to plan the first paragraph of the output. Initially, the most newsworthy message (with highest outlierness value) in the core set is chosen as the *nucleus* (see Section 3.2 for an explanation of terminology) of the first paragraph, establishing the paragraph's theme (consumer price category). Then, the paragraph is completed with *satellite* messages from the union of the core and expanded sets. This set is filtered to include only messages with the same theme. New satellites are appended until the the maximum paragraph length has been reached, or an early-stopping condition is met. The minimum and maximum lengths are three and six messages, respectively. The early stopping condition is met if the highest-scoring satellite candidate falls either below a minimum absolute score threshold, or has a score that is less than 20% of that of the paragraph's nucleus.

Having completed the first paragraph, the second paragraph is initialised by selecting a new nucleus, the message with highest outlierness value not discussing the same theme as the previous one beginning the first paragraph. In this experiment, the evaluated outputs contain two paragraphs each, but the system can be used to generate outputs of arbitrary length.

### Microplanning

Given a document plan, the system proceeds with *microplanning*, including template selection, aggregation, lexicalisation, and referring expression generation. In this stage, messages are associated with linguistic expressions. In our setup, each message is first associated with a short phrase-level template. This is followed by combining templates into longer sentences and replacing parts of the templates with linguistic content.

### Realisation

The final stage of the pipeline, *realisation*, consists of morphological and surface realisation. At this point, the words and phrases constituting a document are inflected into their correct morphological forms, and the text is formatted.

## 6.4.2 Baselines

In terms of document planning, we compare our neural approach, Neural, against two baselines: Trivial and Heuristic. All of these three methods construct paragraphs using fundamentally the same process as described in the previous section. However, they differ in terms of the satellite selection process. In Trivial, the satellites are simply filtered to match the theme of the paragraph's nucleus. That is, a next satellite $s_k$ is the candidate message $c$ for which the score

$$\mathit{score}_T(c, n) = \mathit{outlierness}(c) \times \mathbb{I}(\mathit{theme}(c) = \mathit{theme}(n))$$

where $n$ is the nucleus, is greatest. Content selection and document structuring are then done at the same time, as satellites are selected in the order of descending score values.

Heuristic is effectively the method proposed by Leppänen and Toivonen [84], although without assuming the availability of structural and hierarchical label metadata. We use two weighting schemes, a penalty and a contextual similarity score, which are multiplied with the outlierness value of each message. First, we use a penalty to discourage the generation of long segments of text discussing countries other than the output's focus country given by the user. This is computed as $\mathit{penalty}(c) = \frac{1}{d+1}$, where $d$ is the distance from a candidate satellite $c$ to the paragraph's previous core message. (Core messages always concern the focus country.) That is, $d$ is the number of steps from $c$ to this core message: for example, $d = 0$ if $c$ is itself from the core set, $d = 1$ if the preceding message is from the core set, and so on. This penalty is applied to every candidate in the expanded set.

Second, we compute a contextual similarity score for each candidate satellite, both with regard to the nucleus (first message of the paragraph) and to the previously selected satellite. The similarity score is then the mean of these two components. For the first satellite, the components have the same values. Otherwise, a component's value is 1.5 if the two messages compared have the same timestamp, 2 if they discuss the same country, and 3 if both attributes are the same. If neither is shared, the similarity score is zero. This results in a score for a candidate $c$ to be the next satellite $s_k$ such that

$$score_H(c, n, s_{k-1}) = outlierness(c) \times \frac{sim_c(c, n) + sim_c(c, s_{k-1})}{2} \times penalty(c)$$

where $n$ is the nucleus and $sim_c$ is the contextual similarity metric. The next satellite $s_k$ is the message with the highest score. Thus, content selection and document structuring are done at the same time, as in TRIVIAL.

### 6.4.3 Neural Scoring

In our proposed method, NEURAL, we extend our baseline TRIVIAL by weighting satellite candidates' scores ($score_T$ values defined in the previous section) using our neural sentence ordering model, S-CNN. Our algorithm to do this is presented in Algorithm 1. As in TRIVIAL and HEURISTIC, a paragraph's nucleus is the message in the core set with the highest outlierness value.

To use S-CNN for content selection and document structuring, messages need to be represented as natural-language sentences. To obtain such string-formatted representations, we apply a simplified version of our template-based NLG pipeline to each message in isolation. As we apply it to one message at a time, we do not need steps such as document planning and aggregation that are only meaningful in the context of multiple messages, and thus we ignore them. This also prevents a case where the document planner would have a recursive dependency on itself.

For example, in our use case of consumer price index data, a message containing the information for a country ('Austria'), timestamp ('2021M02'), category ('unprocessed food'), and value for monthly growth rate in comparison with EU average ('5.2%'), the natural language representation would be 'In February 2021, the monthly growth rate of the harmonized consumer price index for the category "unprocessed food" was 5.2 percentage points more than the EU average.' This kind of string representations can be regarded as sentences similar to the actual human-written sentences appearing in our training corpus STATFI.

**Algorithm 1** Algorithm for determining weights $w_N$ with the S-CNN model. The symbol $\oplus$ denotes concatenation.

```
Input: $M = [(m_1, score_T(m_1)), \dots, (m_n, score_T(m_n))]$
Output: $M' = [(m_1, score_N(m_1)), \dots, (m_n, score_N(m_n)]$

$k \leftarrow 20$                                              ▷ Length of adjacent list $J_m$
$a, b \leftarrow [0.02, 50]$                                   ▷ Weight range
$M' = [\ ]$
for all $i \in \{1, \dots, n\}$ do
    # Get adjacent messages
    $i_{\text{left}} \leftarrow \max\{0, i - \lfloor \frac{k}{2} \rfloor\}$        ▷ Left index of $J_m$,
    $i_{\text{right}} \leftarrow \min\{n, i + \lceil \frac{k}{2}$                  ▷ Right index of $J_m$
    $J_m \leftarrow M[i_{\text{left}} : i - 1] \oplus M[i + 1 : i_{\text{right}}]$
    # Compute $score_N$
    $s \leftarrow 0$
    for all $j \in J_m$ do
        $p \leftarrow \text{S-CNN}(m_i, j)$
        $s \leftarrow s + \log p$
    end for
    $s \leftarrow \frac{s}{k-1}$                                ▷ Average $s$ over $J_m$
    $s \leftarrow e^s$
    $w_N \leftarrow s(a - b) + a$                              ▷ Compute weight
    $score_N(m_i) = score_T(m_i) \times w_N$
    $M' \leftarrow M' \oplus [(m_i, score_N(m_i))]$
end for
```

Since S-CNN is trained to classify whether one sentence $s_i$ should precede another sentence $s_j$, we reorder the expanded message set using the model to make pairwise comparisons between messages, according to Algorithm 1. Given the core set as a list of messages $M$ sorted according to decreasing outlierness values, we compute a weight $w_N$ for each message $m \in M$ in the set by comparing it with $k$ adjacent messages $a \in A_m$. We compute the left and right ends of $A$ dividing messages $a$ on both sides of $m$ as evenly as possible, while leaving out $m$ itself, that is $|A_m| = k - 1$. In our experiment, we set $k = 20$. As we select three to six messages out of thousands into a paragraph in the final output, we consider it sufficient to make comparisons between nearest adjacent messages, which also decreases computation time.

Given a message $m$ and its adjacent messages $A_m$, we compute the log-probability sum of S-CNN's predictions $p$ for all pairs between $m$ and $A_m$, and average this value, $s$, over $A_m$. If S-CNN predicts a high value of $s$, then according to the model, $m$ should be placed before the other $k-1$ adjacent messages in $A_m$. We then turn $s$ back to a probability value by taking its exponential, and compute the weight $w_N$ such that $w_N = s(b-a)+a$, where $[a, b]$ is a pre-determined range for the coefficient. The larger this range, the more weight is placed on the neural model's scoring, diminishing the impact of outlierness value. We found a range of [0.02, 50] to be one that weights the neural score quite significantly, resulting in outcomes noticeably different from the baselines (shown later in Figure 6.5). Finally, as a result of Algorithm 1, new satellite candidates $c$ have values

$$score_N(c, n) = score_T(c, n) \times w_N(c)$$

where $score_T(c, n) = outlierness(c) \times \mathbb{I}(theme(c) = theme(n))$, the score of Trivial, and candidate $c$ with the highest value of $score_N$ is selected as the next satellite. Following this scoring procedure, the document planning then proceeds as in the case of Trivial, described above in Section 6.4.2.

### 6.4.4 Human Evaluation

In order to evaluate the news outputs generated using a neural content selection approach, we conducted a questionnaire survey, where human judges were presented with generated texts and a set of statements to be answered using a seven-step Likert scale. Each text was a short two-paragraph news report on consumer price statistics with a focus on one specific European country. In total, each judge evaluated 12 reports generated by three different document planner variants, four reports each, on the same set of four focus countries. We received answers from five judges, resulting in 60 answers per statement, and 20 answers per variant. Example outputs of these three document planner variants (Trivial, Heuristic, Neural) on Finnish consumer prices are shown in Figure 6.5.

The participating judges were journalists from the Finnish News Agency STT as well as doctoral students in journalism, all known to the authors through either a joint research project or through personal connections, but not directly involved with the research described in this paper and were thus volunteers. The participants were not compensated for their work. We assumed our human judges to be quite knowledgeable about two of the four focus countries (Finland and Estonia), while the other two (Croatia and Germany) were most likely unfamiliar to them.

The survey was conducted anonymously online, and the judges were first provided with a brief background to the research and were asked to consent

**Consumer prices in Finland**

```
In March 2020, in Finland, the
monthly growth rate of the
harmonized consumer price index
for the category 'health' was 2.4
points.  In Turkey, the harmonized
consumer price index for the
category 'health' was 70.26 points
more than in US. It was 181.7
points.  In February 2020, it was
65.53 points more than in US. In
March 2020, the monthly growth rate
of the harmonized consumer price
index for the category 'health' was
2.8 points.  In February 2020, the
harmonized consumer price index for
the category 'health' was 176.79
points.

In January 2020, in Finland,
the monthly growth rate of the
harmonized consumer price index
for the category 'education' was 1
percentage points less than in US.
In February 2020, in Turkey, it was
0.9 points.  It was 0.7 percentage
points more than in US. In Sweden,
it was 0.8 points.  It was 0.6
percentage points more than in US.
In January 2020, in Estonia, it was
1.3 percentage points more than in
US.
```

```
In March 2020, in Finland, the
monthly growth rate of the
harmonized consumer price index
for the category 'health' was 2.4
points.  It was 2.2 percentage
points more than in US. In Turkey,
the harmonized consumer price index
for the category 'health' was 70.26
points more than in US. It was 181.7
points.  The monthly growth rate
of the harmonized consumer price
index for the category 'health' was
2.8 points.  Finland had the 2nd
highest monthly growth rate of the
harmonized consumer price index for
the category 'health' across the
observed countries.

In January 2020, the monthly growth
rate of the harmonized consumer
price index for the category
'education' was 1 percentage points
less than in US. It was -0.8 points.
It was 0.8 percentage points less
than the EU average.  In Estonia, it
was 1.3 percentage points more than
in US. It was 1.5 points.  It was
1.5 percentage points more than the
EU average.
```

```
In March 2020, in Finland, the
monthly growth rate of the
harmonized consumer price index
for the category 'health' was 2.4
points.  It was 2.2 percentage
points more than in US. The country
had the 2nd highest monthly growth
rate of the harmonized consumer
price index for the category
'health' across the observed
countries.  In February 2020, the
country had the 12th highest monthly
growth rate of the harmonized
consumer price index for the
category 'health' across the
observed countries.  It was -0.3
points.  It was 0.5 percentage
points less than in US.

In January 2020, the monthly growth
rate of the harmonized consumer
price index for the category
'education' was 1 percentage points
less than in US. It was -0.8 points.
It was 0.8 percentage points less
than the EU average.  In February
2020, it was 0 points.  It was
0.2 percentage points less than
in US. The country had the 8th
highest monthly growth rate of the
harmonized consumer price index for
the category 'education' across the
observed countries.
```

Figure 6.5: Example outputs on Finnish consumer prices. Reports generated by the simple baseline document planner (left), the baseline extended with heuristic fitness scores (middle), and the baseline extended with neural scoring (right).

to the use of their answers in research. They were then provided with a description of the type of reports they would read and other instructions. The texts were described as news alerts, which a journalist would receive from an automated system, and which could be a starting point for a news story. The judges were told that different methods had been used for generation, but they did not know which texts were generated using which method, nor how many methods were being compared. They were shown one output at a time, and asked to indicate their agreement with the following statements:

**Q1:** The text corresponds to the heading (user-queried focus country),

**Q2:** The text is coherent, and

**Q3:** The text contains useful information.

With these statements, we have intended to achieve an understanding of the newsworthiness of the generated texts. Since newsworthiness is not a completely unambiguous concept, we chose three qualities that we consider more straightforward for the judges to relate to: topicality (whether text corresponds to the queried country), coherence, and usefulness. For all of these statements, an answer of 1 indicates complete disagreement while 7 indicates complete agreement on a seven-step Likert scale. We report the median, mean, and standard deviations of the answers for each statement, although we acknowledge that computing mean and standard deviation on a Likert scale is not completely unproblematic as it assumes a unit distance between each step of agreement. However, we think these values do give more insight about the performance differences between the evaluated models.

In order to assess the statistical significance of our results, we conduct first a Kruskal–Wallis test on each statement sample (including all planner variants) to determine whether the samples are significantly different. In the Kruskal-Wallis test, the null hypothesis is that the answers regarding the three variants originate from the same distribution, and its rejection implies there is at least one method whose results are statistically unlikely to result from the same distribution with others. If this null hypothesis is rejected (i.e. $p$-value is below significance level $\alpha$), we apply Mann–Whitney U tests for each pair of methods within a statement sample. The null hypothesis of this test is that the answers regarding the two methods originate from the same distribution. Thus, a rejection of this null hypothesis implies there is likewise a statistically significant difference in performance between the two methods. We use the Kruskal-Wallis test before pairwise Mann–Whitney U tests for additional support to the results of the pairwise tests.

We use an overall significance level of $\alpha_0 = 0.05$ for the Kruskal–Wallis tests, and a Bonferroni-corrected level $\alpha = \frac{\alpha_0}{m} = \frac{0.05}{3} \approx 0.017$ for the Mann–Whitney U tests with $m = 3$ comparisons within each statement sample. We use Bonferroni correction to reduce the possibility of apparent significance arising simply from multiple comparisons.

### 6.4.5 Results

The results of our questionnaire survey are shown in Table 6.3. Altogether, the results indicate that the outputs generated with our neural document planning approach are more topical, coherent, and useful than those of our baselines. he medians and means are higher and standard deviations lower for Neural in comparison with Heuristic and Trivial for all statements Q1–Q3. The scores for Heuristic seem also slightly higher than those of Trivial, although the difference is smaller.

The results of the statistical tests are shown in Table 6.4. The Kruskal–Wallis results indicate that at least one of the methods is significantly different from the others in performance in all statement samples, with a very

| | TRIVIAL | | | HEURISTIC | | | NEURAL | | |
|---|---|---|---|---|---|---|---|---|---|
| | $m$ | $\mu$ | $\sigma$ | $m$ | $\mu$ | $\sigma$ | $m$ | $\mu$ | $\sigma$ |
| Q1 | 4.5 | 4 | 1.97 | 4.5 | 4.65 | 1.27 | 6 | 5.8 | 0.94 |
| Q2 | 3.5 | 3.45 | 1.58 | 4.5 | 4.4 | 1.11 | 5 | 5.15 | 0.94 |
| Q3 | 4 | 3.9 | 1.85 | 4 | 4.55 | 1.28 | 6 | 5.3 | 1.18 |

Table 6.3: The medians ($m$), means ($\mu$), and standard deviations ($\sigma$) of the answers to the questionnaire comparing the outputs generated by document planner variants using the two baselines and our Siamese network with a CNN layer (S-CNN). For each statement, the answer ranges are 1–7, where higher values are better.

| | | $p$-values for statements | | |
|---|---|---|---|---|
| Test | Variants | Q1 | Q2 | Q3 |
| Kruskal–Wallis | All | **0.0026** | **0.0009** | **0.0168** |
| Mann–Whitney U | TRIVIAL – HEURISTIC | 0.1552 | 0.0185 | 0.1082 |
| Mann–Whitney U | NEURAL – TRIVIAL | **0.0010** | **0.0003** | **0.0036** |
| Mann–Whitney U | NEURAL – HEURISTIC | **0.0038** | **0.0161** | 0.0307 |

Table 6.4: $p$-values for the Kruskal–Wallis test for each question sample and all document planner variants, and for the Mann–Whitney U test for each variant pair within each question sample. The significance level of the Kruskal–Wallis test is 0.05, while the Bonferroni-corrected level for Mann–Whitney U tests is 0.017. Statistically significant values are in bold.

high level of confidence for statements Q1 and Q2 ($p < 0.01$ for Q1 and $p < 0.001$ for Q2), and even for Q3, although not quite with the same level of confidence ($p < 0.05$).

The Mann–Whitney U tests for pairs of document planner variants indicate that the difference between the baselines is not statistically significant for any of the three statements ($p \not< 0.017$). This result reflects our observation about the survey results, where the difference between these two methods was relatively small. Meanwhile, the difference between NEURAL and TRIVIAL is statistically significant with high confidence, since $p < 0.01$ for all statements. The same applies to the difference between NEURAL and HEURISTIC, although not with quite as high confidence, and for Q3 the difference is not significant. The above results indicate that NEURAL is clearly superior to TRIVIAL and modestly superior to HEURISTIC.

Altogether, the above results of the survey and the statistical tests indicate that our neural approach to document planning has clearly superior performance in relation to trivial baseline, while also modestly outperforming the heuristic method based on Leppänen and Toivonen [84]. This suggests that the S-CNN is able to learn patterns about the relationship between sentence content and position, and supports our initial assumption that more newsworthy sentences tend to appear earlier than others. In addition, being data-driven, the method requires fewer assumptions to be made about the data.

## 6.5 Conclusion

This chapter has focused on our third research task, RT-III:

**RT-III.** *Given only auxiliary data, develop a method for document planning in news generation.*

We have addressed the task of document planning in a news generation pipeline, considering a truly low-resource scenario with a complete lack of annotated training data. Thus, we could not use direct supervision to train a neural network for the task. Instead, our approach was to use distant supervision, automatically constructing a training dataset for a different but related task, sentence ordering, from auxiliary data in the form of a news corpus. Our work in this chapter provides more insight into the appropriateness of distant supervision in low-resource NLP, an open research question [55].

We have proposed three kinds of neural models for our source task, sentence ordering within paragraphs, representing three lines of work in previous research. First, we considered a Siamese network with either convolutional or recurrent layers to learn pairwise precedence patterns. This model was similar to our cognate identification model in Chapters 4–5. Second, we presented a positional network predicting the quantile a sentence should belong to within a paragraph. Third, we proposed a pointer network, a sequence-to-sequence model using attention to reorder a shuffled set of sentences.

Our experiments were two-fold: first, we evaluated the above neural models on sentence ordering, and we chose one of them to apply to the document planning stage of an existing news generation system [84]. Due to its good performance and relatively lean architecture, we chose to do this using a Siamese network with a convolutional layer (S-CNN). Second, we conducted a human evaluation of news reports generated with different document planner variants, comparing our neural approach with a trivial and a heuristic baseline. Our results showed that the neural document planner brought about superior performance in terms of human-perceived output quality. More specifically, the results imply that its outputs were more topical, coherent, and useful than the baselines.

Altogether, our neural approach provides an interesting avenue for future research on domain-independent methods using distant supervision for NLG, which often involves similar low-resource scenarios regardless of language [57]. Although our results are encouraging, we acknowledge that it has a couple of disadvantages in comparison with heuristic methods [e.g. 83, 84]. First, it is computationally heavier, which might be an issue for online use cases where a user expects fast generation. Second, the method requires a suitable training corpus, the availability of which varies according to domain. Our news corpus, StatFi, was of the same genre of text as in the generation, but including a much wider domain of news topics. This suggests that our approach could potentially transfer well at least within the same genre. Otherwise, the method's performance will most likely be affected by the similarity of chosen training data to the generation domain. Nevertheless, the unlabelled auxiliary data required by our method is relatively easy to gather for new domains.

We conclude that we have accomplished Research Task RT-III by developing a method for document planning in a truly low-resource scenario, using a distantly supervised approach given only unlabelled auxiliary data. Our results provide support for neural transfer learning in the context of higher-level NLP tasks suffering from lack of annotated training data. Being domain-independent, our method is scalable to new domains. Altogether, our work in this chapter contribute to our objective of extending higher-level NLP to truly low-resource domains.

# Chapter 7

# Conclusion

Most of the world's languages do not partake in the recent advancements of NLP, due to insufficient data resources for deep learning with neural networks. In addition, even some domains and tasks face a similar situation, regardless of language. Indeed, truly low-resource languages might lack even basic NLP tools, and extremely low-resource domains any annotated data whatsoever for certain tasks, even within a high-resource language.

The purpose of this thesis has been to support the extension of basic NLP to a wider range of truly low-resource languages, and the coverage of higher-level NLP across low-resource domains. We have attempted to do this by addressing two different tasks. On the one hand, we focused on cognate identification, since cognate information facilitates cross-lingual transfer for low-level NLP tasks. On the other hand, we examined the higher-level task of document planning, fundamental in NLG and more advanced NLP applications, but still largely dependent on domain-specific heuristics.

We have considered these tasks in three kinds of low-resource scenarios with varying resource conditions. First, we identified cognates between endangered Sami languages, given only a small amount of labelled data in these languages. Second, we considered the same task in a scenario where target-language data is completely unlabelled. Third, we addressed document planning for news generation in a domain without annotated training data. Since direct supervision of neural networks has been unfeasible in these scenarios, we have proposed several transfer learning approaches.

Next, we present an overview of our contributions with regard to our three research tasks, followed by discussions of both our results and future research avenues.

## 7.1 Contributions

Our objective in this thesis has been to develop neural-network solutions to cognate identification and document planning, and to use transfer learning to adapt them to three different low-resource scenarios. To meet this objective, we defined one research task (RT) for each low-resource scenario in Chapter 1. In each of Chapters 4–6, we have addressed one RT.

**Research Task I:**

*Given scarce language-specific labelled data, identify cognates in truly low-resource Sami languages.*

We addressed this research task (RT-I) in Chapter 4. We focused on the task of cognate identification between truly low-resource South, North, and Skolt Sami of the Uralic family. In this low-resource scenario, available training data consisted of a small cognate set, insufficient for direct supervised learning. Lacking high-resource close relatives, we chose the highly unrelated Indo-European language family as our source language set.

We proposed two similarity learning models: a support vector machine (SVM) with string-metric features, and a Siamese convolutional neural network (S-CNN). We pre-trained these using Indo-European cognate sets, and found that S-CNN generalised better to Sami without any adaptation. We also analysed how effectively it would adapt when fine-tuned with a Sami cognate set, and found that it achieved noticeable improvements already with small amounts of fine-tuning samples. In contrast to most previous work where source and target languages are close relatives, we consider a novel setup where they are unrelated. The good performance of S-CNN implies that it can learn such patterns of cognacy that carry over from one language family to another.

**Research Task II:**

*Given only unlabelled data, adapt a pre-trained cognate identification model to truly low-resource Uralic languages.*

This research task (RT-II) was the focus of Chapter 5. While continuing with the task of cognate identification, we considered a novel scenario where our target languages were low-resource to the extent that no labelled data was available for fine-tuning. Again, we used Indo-European source data and Uralic (Sami and Finnic) target data.

We proposed two unsupervised approaches: domain adaptation with discriminative adversarial networks, and cross-lingual adaptation using pre-trained cross-lingual symbol embeddings. These were both based on adapting the model's representations, although at different levels. Neither had been used before for cognate identification. We found that an adversarially adapted model outperformed an unadapted one at identifying Sami cognates. While symbol embeddings did not improve S-CNN's performance at identifying Finnic cognates, we concluded that they would likely be beneficial in scenarios where languages have disparate orthographies.

**Research Task III:**

*Given only auxiliary data, develop a method for document planning in news generation.*

This research task was addressed in Chapter 6. We examined the task of content selection and ordering, known as document planning, in the generation of statistical news reports. In this previously unaddressed low-resource scenario, although generating in English, we lacked suitably annotated data for direct supervised training. Therefore, we made use of auxiliary data in the form of a news corpus.

We proposed a novel method of distant supervision: using our auxiliary data, we automatically constructed labelled training data for the task of sentence ordering, our source task, related to document planning, our target task. We compared Siamese, positional, and pointer networks at sentence ordering, and found a variant of the Siamese convolutional network (S-CNN) most suitable for document planning. Based on human evaluation, we found that using this model improved generated outputs in comparison with heuristic baselines.

## 7.2 Discussion

The results of our research tasks imply that neural networks provide promising solutions to our tasks even in truly low-resource scenarios, when leveraged with transfer learning. In particular, we have found that Siamese convolutional networks have been effective at learning cognacy between words, as well as relative order between sentences. Lacking training data for direct supervised learning, we have used fine-tuning, unsupervised cross-lingual adaptation, and distant supervision to transfer our neural models to our low-resource scenarios.

For both of our tasks, our proposed models and transfer learning approaches have been language- and domain-independent. Thus, we consider our methods scalable to new low-resource scenarios with different source and target languages or domains. We have demonstrated that a language-independent neural model, in combination with fine-tuning or unsupervised adaptation, can identify cognates in truly low-resource languages when pre-trained on an unrelated language family. Likewise, we developed a similar domain-independent neural approach to document planning, trained only on auxiliary data. This indicates that our contributions support the overall expansion of NLP to a larger set of languages and domains that are currently truly low-resource. Since cognate information is useful for cross-lingual transfer, new methods for cognate identification are in themselves valuable for low-resource NLP.

Our results have interesting implications for both historical linguistics and data-to-text NLG. Transferring a cognate identification model from one language family to another implies that predictors of cognacy, such as regular sound correspondences, carry over across language families. This finding opens up an interesting avenue for studying universal patterns of sound change, which might be of interest for the fields of historical linguistics, phonetics, and linguistic typology.

Our results also indicate the overlap between NLP and other areas of machine learning, as well as within NLP. We have found convolutional networks especially useful for our tasks, although they are most prominent in computer vision. In addition, we found our adversarial adaptation method, originally designed for image domains, to perform well at cognate identification. Yet another example of this is the use of distant supervision, popular in image classification [55], for document planning. We also show that cross-lingual symbol embeddings are beneficial for cognate identification and probably other cross-lingual lower-level tasks, analogously to word embeddings so crucial for higher-level NLP.

Although our methods are more scalable than heuristics and rule-based methods requiring language- or domain-expertise, they require certain kinds of training data in varying amounts. In some truly low-resource scenarios, even unlabelled data might be too scarce for our methods. However, this kind of data is relatively easy to gather, as no annotations are needed. In addition, our methods are computationally more demanding than heuristic algorithms, which could be an issue especially in the case of document planning and online generation.

.

## 7.3 Future Work

While our findings in this thesis have been encouraging for neural transfer learning in truly low-resource NLP, we acknowledge that they provide only a small step towards a more general coverage of NLP across languages and domains.

In the context of cognate identification, we have shown that cross-lingual transfer is possible from one language family to another. This finding deviates from most previous work in cross-lingual transfer considering only close relatives, thus opening up an interesting avenue for future work in low-resource NLP. It would be important to examine how universal our result is, by varying the degree of relatedness of source and target languages used for pre-training and testing, considering transfer both within and between different language families. Moreover, some of our methods could be used in combination, especially those of unsupervised adaptation.

The setup of transfer learning between language families could possibly be extended to other tasks as well. The characteristics of NLP tasks suitable for such distant transfer could be investigated, and the extent of language relatedness needed by different tasks analysed. We speculate that distant transfer is more feasible for tasks where language is represented on the level of symbols or sounds, as in our cognate identification task. Examples of such applications could be related to phonetics, such as speech recognition [131].

In terms of low-resource domains, we have demonstrated the benefit of distant supervision in the context of document planning for news generation. While we consider this an interesting result, it would be important to continue this work with additional experiments considering different domains (e.g. news genres) and more dissimilar auxiliary datasets. Furthermore, the impact of the amount of auxiliary data should be investigated, to evaluate the feasibility of the method in lower-resource domains and languages.

Although our methods are language- and domain-independent, they might not be applicable to all low-resource scenarios as they are. Therefore, another line of future research would be to investigate combinations of our methods with rule-based ones relying on language- or domain-expertise. As indicated by other recent work [149], this is likely the way to go for low-resource NLP.

# References

[1] Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. Cross-lingual word embeddings for low-resource language modeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 937–947. Association for Computational Linguistics, April 2017.

[2] Harsh Agrawal, Arjun Chandrasekaran, Dhruv Batra, Devi Parikh, and Mohit Bansal. Sort story: Sorting jumbled images and captions into stories. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 925–931, Austin, Texas, November 2016. Association for Computational Linguistics.

[3] Firoj Alam, Shafiq Joty, and Muhammad Imran. Domain adaptation with adversarial training and graph embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1077–1087, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[4] Hanan Aldarmaki and Mona Diab. Scalable cross-lingual transfer of neural sentence embeddings. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 51–60, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[5] Gabor Angeli, Percy Liang, and Dan Klein. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512, Cambridge, MA, October 2010. Association for Computational Linguistics.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer

normalization. In *Advances in NIPS 2016: Deep Learning Symposium*, 2016.

[7] Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, California, May 2015.

[8] Hadi Banaee, Mobyen Uddin Ahmed, and Amy Loutfi. Towards NLG for physiological data monitoring with body area networks. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 193–197, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[9] Khuyagbaatar Batsuren, Gábor Bella, and Fausto Giunchiglia. A large and evolving cognate database. *Language Resources and Evaluation*, 56(1):165–189, March 2022.

[10] Lisa Beinborn, Torsten Zesch, and Iryna Gurevych. Cognate production using character-based machine translation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 883–891, Nagoya, Japan, October 2013. Asian Federation of Natural Language Processing.

[11] Lisa Beinborn, Torsten Zesch, and Iryna Gurevych. Readability for foreign language learning: The importance of cognates. *ITL-International Journal of Applied Linguistics*, 165(2):136–162, 2014.

[12] Shane Bergsma and Grzegorz Kondrak. Alignment-based discriminative string similarity. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 656–663, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[13] Michael Bloodgood and Benjamin Strauss. Using global constraints and reranking to improve cognates detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1983–1992, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[14] Tanner Bohn, Yining Hu, Jinhang Zhang, and Charles Ling. Learning sentence embeddings for coherence modelling and beyond. In *Proceedings of the International Conference on Recent Advances in Natural*

*Language Processing (RANLP 2019)*, pages 151–160, Varna, Bulgaria, September 2019. INCOMA Ltd.

[15] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[16] Sidsel Boldsen, Manex Agirrezabal, and Nora Hollenstein. Interpreting character embeddings with perceptual representations: The case of shape, sound, and color. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6819–6836, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[17] Alexandre Bouchard-Côté, David Hall, Thomas L. Griffiths, and Dan Klein. Automated reconstruction of ancient languages using probabilistic models of sound change. *Proceedings of the National Academy of Sciences*, 110(11):4224–4229, 2013.

[18] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, ..., and Dario Amodei. Language models are few-shot learners. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1877–1901, Online, 2020. Curran Associates, Inc.

[19] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150, Munich, Germany, September 2018. Springer.

[20] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[21] Will Chang, Chundra Cathcart, David Hall, and Andrew Garrett. Ancestry-constrained phylogenetic analysis supports the Indo-European steppe hypothesis. *Language*, 91(1):194–244, 2015.

[22] Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online, November 2020. Association for Computational Linguistics.

[23] Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570, 2018.

[24] Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Neural sentence ordering. *arXiv preprint arXiv:1607.06952*, 2016.

[25] François Chollet et al. Keras. Technical report, Github, 2015.

[26] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 539–546, San Diego, California, 2005. IEEE.

[27] Alina Maria Ciobanu and Liviu P. Dinu. Automatic detection of cognates using orthographic alignment. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 99–105, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[28] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics.

[29] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[30] Baiyun Cui, Yingming Li, Yaqing Zhang, and Zhongfei Zhang. Text coherence analysis based on deep neural network. In *Proceedings of the 2017 ACM Conference on Information and Knowledge Management*

*(CIKM '17)*, page 2027–2030, New York, New York, 2017. Association for Computing Machinery.

[31] Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[33] C. Downey, Shannon Drizin, Levon Haroutunian, and Shivin Thukral. Multilingual unsupervised sequence segmentation transfers to extremely low-resource languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5331–5346, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[34] Lukas Edman, Antonio Toral, and Gertjan van Noord. The importance of context in very low resource language modeling. In *Proceedings of the 18th International Conference on Natural Language Processing (ICON)*, pages 86–92, National Institute of Technology Silchar, Silchar, India, December 2021. NLP Association of India.

[35] Ramy Eskander, Nizar Habash, Owen Rambow, and Nadi Tomeh. Processing spontaneous orthography. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 585–595, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[36] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):2096–2030, January 2016.

[37] Albert Gatt and Emiel Krahmer. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170, 2018.

[38] Albert Gatt, Francois Portet, Ehud Reiter, Jim Hunter, Saad Mahamood, Wendy Moncur, and Somayajulu Sripada. From data to text in the neonatal intensive care unit: Using NLG technology for decision support and information management. *AI Communications*, 22(3):153–186, 2009.

[39] Gerard de Melo. Etymological WordNet: Tracing the History of Words. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[40] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*, page 513–520, Madison, Wisconsin, 2011. Omnipress.

[41] Luís Gomes and José Gabriel Pereira Lopes. Measuring spelling similarity for cognate identification. In *Proceedings of the 15th Portugues Conference on Artificial Intelligence*, pages 624–633, Lisbon, Portugal, 2011. Springer Berlin Heidelberg.

[42] Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. End-to-end neural sentence ordering using pointer network. *arXiv preprint arXiv:1611.04953*, 2016.

[43] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[44] Mark Granroth-Wilding and Hannu Toivonen. Unsupervised learning of cross-lingual symbol embeddings without parallel data. In *Proceedings of the Society for Computation in Linguistics*, volume 2, pages 19–28, New York, New York, 2019.

[45] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

[46] Andreas Grivas, Beatrice Alex, Claire Grover, Richard Tobin, and William Whiteley. Not a cute stroke: Analysis of rule- and neural network-based information extraction systems for brain radiology reports. In *Proceedings of the 11th International Workshop on Health*

*Text Mining and Information Analysis*, pages 24–37, Online, November 2020. Association for Computational Linguistics.

[47] Stig-Arne Grönroos, Sami Virpioja, and Mikko Kurimo. Cognate-aware morphological segmentation for multilingual neural translation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 386–393, Belgium, Brussels, October 2018. Association for Computational Linguistics.

[48] Grigorii Guz and Giuseppe Carenini. Towards domain-independent text structuring trainable on large discourse treebanks. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3141–3152, Online, November 2020. Association for Computational Linguistics.

[49] Mika Hämäläinen. Endangered languages are not low-resourced! In Mika Hämäläinen, Niko Partanen, and Khalid Alnajjar, editors, *Multilingual Facilitation*, pages 1–11. University of Helsinki, March 2021.

[50] Mika Hämäläinen and Jack Rueter. Finding Sami cognates with a character-based NMT approach. In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages: Volume 1 (Papers)*, pages 39–45, Honolulu, February 2019. Association for Computational Linguistics.

[51] Bradley Hauer, Amir Ahmad Habibi, Yixing Luan, Rashed Rubby Riyadh, and Grzegorz Kondrak. Cognate projection for low-resource inflection generation. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 6–11, Florence, Italy, August 2019. Association for Computational Linguistics.

[52] Bradley Hauer and Grzegorz Kondrak. Clustering semantically equivalent words into cognate sets in multilingual lists. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 865–873, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

[53] Bradley Hauer, Garrett Nicolai, and Grzegorz Kondrak. Bootstrapping unsupervised bilingual lexicon induction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 619–624, Valencia, Spain, April 2017. Association for Computational Linguistics.

[54] Michael A. Hedderich, David Adelani, Dawei Zhu, Jesujoba Alabi, Udia Markus, and Dietrich Klakow. Transfer learning and distant supervision for multilingual transformer models: A study on African languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2580–2591, Online, November 2020. Association for Computational Linguistics.

[55] Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online, June 2021. Association for Computational Linguistics.

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

[57] David M. Howcroft and Dimitra Gkatzia. Most NLG is low-resource: here's what we can do about it. In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 336–350, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics.

[58] Tsung-Yuan Hsu, Chi-Liang Liu, and Hung-yi Lee. Zero-shot reading comprehension by cross-lingual transfer learning with multi-lingual language representation model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5933–5940, Hong Kong, China, November 2019. Association for Computational Linguistics.

[59] Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR, July 2020.

[60] Patrick Huber and Giuseppe Carenini. MEGA RST discourse treebanks with structure and nuclearity from scalable distant sentiment supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7442–7457, Online, November 2020. Association for Computational Linguistics.

[61] Institute for the Languages of Finland. Álgu database. Sámegielaid etymologaš diehtovuođđu – etymological database of the Sami languages. `http://https://kaino.kotus.fi/algu/`, November 2006.

[62] Gerhard Jäger, Johann-Mattis List, and Pavel Sofroniev. Using support vector machines and state-of-the-art algorithms for phonetic alignment to identify cognates in multi-lingual wordlists. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1205–1216, Valencia, Spain, April 2017. Association for Computational Linguistics.

[63] Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online, July 2020. Association for Computational Linguistics.

[64] Gerhard Jäger. Phylogenetic inference from word lists using weighted alignment with empirically determined weights. *Language Dynamics and Change*, 3(2):245 – 291, 2013.

[65] Anush Kamath, Sparsh Gupta, and Vitor Carvalho. Reversing gradients in adversarial domain adaptation for question deduplication and textual entailment tasks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5545–5550, Florence, Italy, July 2019. Association for Computational Linguistics.

[66] Diptesh Kanojia, Raj Dabre, Shubham Dewangan, Pushpak Bhattacharyya, Gholamreza Haffari, and Malhar Kulkarni. Harnessing cross-lingual features to improve cognate detection for low-resource languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1384–1395, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[67] Diptesh Kanojia, Sravan Munukutla, Sayali Ghodekar, Pushpak Bhattacharyya, and Malhar Kulkarni. Keep your dimensions on a leash: True cognate detection using siamese deep neural networks. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data (CoDS-COMAD) 2020 (7th ACM IKDD CoDS and 25th COMAD)*, page 324–325, Hyderabad, India, 2020. Association for Computing Machinery.

[68] Joo-Kyung Kim, Young-Bum Kim, Ruhi Sarikaya, and Eric Fosler-Lussier. Cross-lingual transfer learning for POS tagging without cross-lingual resources. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2832–2838, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[69] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[70] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2741–2749, Phoenix, California, February 2016. AAAI Press.

[71] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, California, May 2015.

[72] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *Proceedings of the 32nd International Conference on Machine Learning: Deep Learning Workshop*, volume 2, 2015.

[73] Grzegorz Kondrak. A new algorithm for the alignment of phonetic sequences. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 288–295, Seattle, Washington, 2000. Association for Computational Linguistics.

[74] Grzegorz Kondrak. Identifying cognates by phonetic and semantic similarity. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, pages 1–8, Pittsburgh, Pennsylvania, 2001. Association for Computational Linguistics.

[75] Grzegorz Kondrak. N-gram similarity and distance. In *Proceedings of the 12th International Symposium on String Processing and Information Retrieval*, pages 115–126, Buenos Aires, Argentina, 2005. Springer Berlin Heidelberg.

[76] Grzegorz Kondrak. Identification of cognates and recurrent sound correspondences in word lists. *Traitement Automatique des Langues (TAL)*, 50(2):201–235, 2009.

[77] Anirban Laha, Parag Jain, Abhijit Mishra, and Karthik Sankaranarayanan. Scalable micro-planned generation of discourse from structured data. *Computational Linguistics*, 45(4):737–763, 2020.

[78] Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, BC, Canada, April–May 2018. OpenReview.net.

[79] Lukas Lange, Heike Adel, and Jannik Strötgen. On the choice of auxiliary languages for improved sequence tagging. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 95–102, Online, July 2020. Association for Computational Linguistics.

[80] Mirella Lapata. Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484, 2006.

[81] Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online, November 2020. Association for Computational Linguistics.

[82] Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378, 2017.

[83] Leo Leppänen, Myriam Munezero, Stefanie Sirén-Heikel, Mark Granroth-Wilding, and Hannu Toivonen. Finding and expressing news from structured data. In *Proceedings of the 21st International Academic Mindtrek Conference*, page 174–183, Tampere, Finland, 2017. Association for Computing Machinery.

[84] Leo Leppänen and Hannu Toivonen. A baseline document planning method for automated journalism. In *Proceedings of the 23rd Nordic*

*Conference on Computational Linguistics (NoDaLiDa)*, pages 101–111, Online, May–June 2021. Linköping University Electronic Press, Sweden.

[85] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[86] Jiwei Li and Dan Jurafsky. Neural net models of open-domain discourse coherence. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 198–209, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[87] Xiangyang Li, Xiang Long, Yu Xia, and Sujian Li. Low resource style transfer via domain adaptive meta learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3014–3026, Seattle, Washington, July 2022. Association for Computational Linguistics.

[88] Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy, July 2019. Association for Computational Linguistics.

[89] Johann-Mattis List. *Sequence comparison in historical linguistics.* PhD thesis, Heinrich-Heine-Universität Düsseldorf, 2013.

[90] Qianchu Liu, Diana McCarthy, Ivan Vulić, and Anna Korhonen. Investigating cross-lingual alignment methods for contextualized embeddings with token-level evaluation. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 33–43, Hong Kong, China, November 2019. Association for Computational Linguistics.

[91] Lajanugen Logeswaran, Honglak Lee, and Dragomir Radev. Sentence ordering and coherence modeling using recurrent neural networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5285–5292. AAAI Press, 2018.

[92] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[93] William C Mann and Sandra A Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988.

[94] Richard T. McCoy and Robert Frank. Phonologically informed edit distance algorithms for word alignment with low-resource languages. In *Proceedings of the Society for Computation in Linguistics*, volume 1, pages 102–112, Salt Lake City, Utah, 2018.

[95] Oren Melamud, Mihaela Bornea, and Ken Barker. Combining unsupervised pre-training and annotator rationales to improve low-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3884–3893, Hong Kong, China, November 2019. Association for Computational Linguistics.

[96] Vladislav Mikhailov, Lorenzo Tosi, Anastasia Khorosheva, and Oleg Serikov. Initial experiments in cross-lingual morphological analysis using morpheme segmentation. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 144–152, Ann Arbor, Michigan, June 2019. Association for Computational Linguistics.

[97] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, volume 2, pages 3111–3119, Lake Tahoe, Nevada, 2013. Curran Associates, Inc.

[98] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[99] Christopher Moseley. *Atlas of the World's Languages in Danger*. UNESCO Publishing, 3rd edition, 2010. `http://www.unesco.org/languages-atlas`.

[100] Preslav Nakov and Jörg Tiedemann. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 301–305, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[101] Christopher Norman, Mariska Leeflang, René Spijker, Evangelos Kanoulas, and Aurélie Névéol. A distantly supervised dataset for automated data extraction from diagnostic studies. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 105–114, Florence, Italy, August 2019. Association for Computational Linguistics.

[102] Byungkook Oh, Seungmin Seo, Cheolheon Shin, Eunju Jo, and Kyong-Ho Lee. Topic-guided coherence modeling for sentence ordering by preserving global and local information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2273–2283, Hong Kong, China, November 2019. Association for Computational Linguistics.

[103] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[104] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 8026–8037, Vancouver, Canada, 2019. Curran Associates, Inc.

[105] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and

E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[106] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 3934–3941, New Orleans, Louisiana, April 2018. AAAI Press.

[107] 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, Doha, Qatar, October 2014. Association for Computational Linguistics.

[108] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[109] Barbara Plank. *Domain Adaptation for Parsing.* PhD thesis, University of Groningen, 2011.

[110] Barbara Plank and Željko Agić. Distant supervision from disparate sources for low-resource part-of-speech tagging. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 614–620, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[111] Barbara Plank, Anders Søgaard, and Yoav Goldberg. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany, August 2016. Association for Computational Linguistics.

[112] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[113] Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W. Black. Topological sort for sentence ordering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2783–2792, Online, July 2020. Association for Computational Linguistics.

[114] Ratish Puduppully, Li Dong, and Mirella Lapata. Data-to-text generation with content selection and planning. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 6908–6915, Honolulu, Hawaii, July 2019. AAAI Press.

[115] Ratish Puduppully and Mirella Lapata. Data-to-text generation with macro planning. *Transactions of the Association for Computational Linguistics*, 9:510–527, 2021.

[116] Ratish Surendran Puduppully. *Data-to-text generation with neural planning*. PhD thesis, University of Edinburgh, 2022.

[117] Taraka Rama. Automatic cognate identification with gap-weighted string subsequences. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1227–1231, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[118] Taraka Rama. Siamese convolutional networks for cognate identification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1018–1027, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[119] Alan Ramponi and Barbara Plank. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[120] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics.

[121] Ehud Reiter and Rober Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.

[122] Sebastian Ruder. The 4 biggest open problems in NLP. *Ruder.io*. `https://www.ruder.io/4-biggest-open-problems-in-nlp/`, 2019.

[123] Sebastian Ruder. *Neural Transfer Learning for Natural Language Processing*. PhD thesis, National University of Ireland, Galway, 2019.

[124] Sebastian Ruder, Ivan Vulić, and Anders Søgaard. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631, 2019.

[125] Yves Scherrer and Benoît Sagot. A language-independent and fully unsupervised approach to lexicon induction and part-of-speech tagging for closely related languages. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 502–508, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

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

[127] Darsh Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. Adversarial domain adaptation for duplicate question detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1056–1063, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[128] Aili Shen and Timothy Baldwin. A simple yet effective method for sentence ordering. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 154–160, Singapore and Online, July 2021. Association for Computational Linguistics.

[129] Joongbo Shin, Yanghoon Kim, Seunghyun Yoon, and Kyomin Jung. Contextual-CNN: A novel architecture capturing unified meaning for sentence classification. In *Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 491–494, Shanghai, China, January 2018. IEEE.

[130] Aditya Siddhant, Preethi Jyothi, and Sriram Ganapathy. Leveraging native language speech for accent identification using deep siamese networks. In *Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 621–628, Okinawa, Japan, December 2017. IEEE.

[131] Mittul Singh, Peter Smit, Sami Virpioja, and Mikko Kurimo. Effects of language relatedness for cross-lingual transfer learning in character-based language models. In *Proceedings of the First Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 41–45, Marseille, France, May 2020. European Language Resources Association.

[132] Eliel Soisalon-Soininen and Mark Granroth-Wilding. Cross-family similarity learning for cognate identification in low-resource languages. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1121–1130, Varna, Bulgaria, September 2019. INCOMA Ltd.

[133] Eliel Soisalon-Soininen and Mark Granroth-Wilding. Transfer learning for cognate identification in low-resource languages. In *Proceedings of TyP-NLP: The First Workshop on Typology for Polyglot NLP (Abstracts)*, 2019.

[134] Eliel Soisalon-Soininen and Mika Hämäläinen. Automated cognate discovery in the context of low-resource Sami languages. In *Proceedings of the Digital Humanities in the Nordic Countries 3rd Conference (Abstracts)*, 2018.

[135] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[136] Adam St Arnaud, David Beck, and Grzegorz Kondrak. Identifying cognate sets across dictionaries of related languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2519–2528, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[137] Craig Thomson, Ehud Reiter, and Somayajulu Sripada. Comprehension driven document planning in natural language generation systems. In *Proceedings of the 11th International Conference on Natural*

*Language Generation*, pages 371–380, Tilburg University, The Netherlands, November 2018. Association for Computational Linguistics.

[138] Elizabeth A. Thomson, Peter R. R. White, and Philip Kitley. "Objectivity" and "hard news" reporting across cultures. *Journalism Studies*, 9(2):212–228, 2008.

[139] Yulia Tsvetkov and Chris Dyer. Cross-lingual bridges with models of lexical borrowing. *Journal of Artificial Intelligence Research (JAIR)*, 55:63–93, 2016.

[140] Yulia Tsvetkov, Sunayana Sitaram, Manaal Faruqui, Guillaume Lample, Patrick Littell, David Mortensen, Alan W Black, Lori Levin, and Chris Dyer. Polyglot neural language models: A case study in cross-lingual phonetic representation learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1357–1366, San Diego, California, June 2016. Association for Computational Linguistics.

[141] Peter Turchin, Ilia Peiros, and Gell-Mann Murray. Analyzing genetic connections between languages by matching consonant classes. *Journal of Language Relationship*, 3:117–126, 2010.

[142] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2962–2971, Honolulu, Hawaii, 2017. IEEE.

[143] Ahmet Üstün, Gosse Bouma, and Gertjan van Noord. Cross-lingual word embeddings for morphologically rich languages. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1222–1228, Varna, Bulgaria, September 2019. INCOMA Ltd.

[144] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 5998–6008, Long Beach, California, 2017. Curran Associates, Inc.

[145] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Proceedings of the 28th International Conference on Neural*

*Information Processing Systems*, pages 2692–2700, Montréal, Canada, 2015. Curran Associates, Inc.

[146] Runchuan Wang, Zhao Zhang, Fuzhen Zhuang, Dehong Gao, Yi Wei, and Qing He. Adversarial domain adaptation for cross-lingual information retrieval with multilingual bert. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, CIKM '21, page 3498–3502, Virtual Event, Queensland, Australia, 2021. Association for Computing Machinery.

[147] Tianming Wang and Xiaojun Wan. Hierarchical attention networks for sentence ordering. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 7184–7191, Honolulu, Hawaii, July 2019. AAAI Press.

[148] Peter White. Death, disruption and the moral order: the narrative impulse in mass-media 'hard news' reporting. *Genres and institutions: Social processes in the workplace and school*, 101:133, 1997.

[149] Linda Wiechetek, Flammie Pirinen, Mika Hämäläinen, and Chiara Argese. Rules ruling neural networks - neural vs. rule-based grammar checking for a low resource language. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1526–1535, Online, September 2021. INCOMA Ltd.

[150] Sam Wiseman, Stuart Shieber, and Alexander Rush. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP*, pages 2253–2263, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[151] Shiyao Xu, Shijia E, and Yang Xiang. Enhanced attentive convolutional neural networks for sentence pair modeling. *Expert Systems with Applications*, 151:113384, 2020.

[152] Weijia Xu and Marine Carpuat. Rule-based morphological inflection improves neural terminology translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5902–5914, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[153] Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. Distantly supervised NER with partial annotation learning

and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2159–2169, Santa Fe, New Mexico, August 2018. Association for Computational Linguistics.

[154] Xunjian Yin and Xiaojun Wan. How do Seq2Seq models perform on end-to-end data-to-text generation? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7701–7710, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[155] Yongjing Yin, Shaopeng Lai, Linfeng Song, Chulun Zhou, Xianpei Han, Junfeng Yao, and Jinsong Su. An external knowledge enhanced graph-based neural network for sentence ordering. *Journal of Artificial Intelligence Research*, 70:545–566, 2021.

[156] Yongjing Yin, Linfeng Song, Jinsong Su, Jiali Zeng, Chulun Zhou, and Jiebo Luo. Graph-based neural sentence ordering. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5387–5393, Macao, China, July 2019. International Joint Conferences on Artificial Intelligence Organization.

[157] Matthew D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[158] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 649–657, Montréal, Canada, 2015. Curran Associates, Inc.

[159] Joey Zhou, Sinno Pan, Ivor Tsang, and Yan Yan. Hybrid heterogeneous transfer learning through deep learning. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 2213–2219, Québec, Canada, June 2014. AAAI Press.