D1.1.4 & 5.2: Semantic content matching
Module specification & Semantic search interface

Authors:

Björn Gambäck
NTNU
`gamback@idi.ntnu.no`

Utpal Kumar Sikdar
NTNU
`utpals@idi.ntnu.no`

Lars Bungum
NTNU
`larsbun@idi.ntnu.no`

Grant Agreement Number	7F14047
Project Acronym	HaBiT
Project Title	Harvesting big text data for under-resourced languages
Deliverable Title	**D1.1.4 & 5.2: Semantic content matching** **Module specification & Semantic search interface**
Responsible Partner	**Björn Gambäck, NTNU**
Dissemination Level	Public
Due Delivery Date	31 December 2016 (+30 days)
Actual Delivery Date	15 January 2017
Status	Final, v1.0

Principal Investigator	**Karel Pala**
Project Promoter	**Masaryk University**
Tel	+420 549 49 5616
E-mail	`pala@fi.muni.cz`
Project Website Address	`www.habit-project.eu`

Table of Contents

Chapter 1

Introduction

The main task of WP5 is to increase the usability of the HaBiT resources through semantic search options, and doing this by investigating word-level semantic matching and disambiguation. Semantic search has an important part in improving search accuracy by understanding the intention of the search user. The contextual information carries effective meaning where the search term appears in the text. There are several aspects of this that could and should be investigated, in particular when in the web page and social media text tyoe setting that the HaBiT system is facing.

First, it is necessary to indentify what actually constitutes different domains in these types of texts. That two texts come from social media does not in itself imply that they belong to *one*, delimited textual domain. Rather, there is a wide spectrum of different types of texts that are transmitted through social media, and the level of formality of the language in addition depends more on the style of the writer than on the actual media (Eisenstein, 2013; Androutsopoulos, 2011). They both argue that the common denominator of social media text is not that it is 'noisy' and informal *per se*, but that it describes language in (rapid) change. Furthermore, although social media often convey more ungrammatical text than more formal writings, Baldwin et al. (2013) have shown that the relative occurrence of non-standard syntax is fairly constant among many types of media, such as mails, tweets, forums, comments, and blogs. Hence Chapter 2 starts out with a discussion of domains and domain adaptation.

Second, it is not altogether straight-forward to establish what language different texts are written in. If it is possible to identify the terms that belong to a particular language, it will be more easy to carry out semantic search in the context of a particular text. In particular, texts on the web and social media tend to exhibit a higher level of code-switching and code-mixing than more formal text, so correct language identification in code-switched text aids the semantic search. When two individuals who are bi- or multi-lingual in an overlapping set of languages communicate, they tend to switch seemlessly and effortlessly between the languages (codes) they share. When this code alternation occurs at or above the utterance level, the phenomenon is referred to as code-switching; when the alternation is utterance-internal, the term 'code-mixing' is common, even though 'code-switching' is frequently used in those cases as well. Code-mixing in itself is often an effect of what recently (particularly within language teaching) has started to be called 'translanguaging', that is, when truly bilingual individuals are creating new meanings based on their full and double language repertoire (Lewis et al., 2012). Code-switching is most prominent in spoken language conversations and has thus traditionally mainly been studied by psycho- and sociolinguists (Auer, 1999; Muysken, 2000; Gafaranga and Torras, 2002; Bullock et al., 2014) and by speech researchers (Lyu et al., 2015), while the lack of large-scale textual corpora has made code-switching less attractive as a subject of study in computational or corpora linguistics. However, this started changing in 2003 with the advent of social media, where large amounts of texts are written that are more informal and more conversational in nature, and hence when produced by bilinguals tend to contain more code-switching Paolillo (1996). Language identification in code-switched text is thus the topic for Chapter 3.

Third, it is thus a major issue to decide on what texts and texts types to concentrate on. Due to the ease of availability of Twitter, most research on social media text has so far focused on tweets (Twitter messages). Lui and Baldwin (2014) note that users that mix languages in their writing still tend to avoid code-switching inside a specific tweet, a fact that has been utilized to investigate which language is dominant in a tweet (Carter, 2012; Lignos and Marcus, 2013; Voss et al., 2014). However, tweets still tend to be somewhat formal by more often following grammatical norms and using standard lexical items (Hu et al., 2013), while chats are more conversational (Paolillo, 1999), and hence less formal, which has several side-effects, such as increasing the level of code-switching (Cárdenas-Claros and Isharyanti, 2009; Paolillo, 2011; Nguyen and Doğruöz, 2013; Das and Gambäck, 2014).

Fourth, word-level semantic matching and disambiguation in itself entails a wide range of possible tasks. The original HaBiT project proposal suggested the possibility to experiment with word sense disambiguation, and in patricular stated (p.13) that we would aim to use Multi-Sense Random Indexing (Moen et al., 2013), a technique based on Random Indexing (Kanerva et al., 2000). However, the availability of test and training data made it more relevant to look at a related word-level disambiguating task, namely named entity recognition. Named entity recognition would play an important role to help the semantic search in a more efficient way. For example, if the search term is 'apple', it may refer to a common noun or proper noun (the company). If the user intention of the search term is only the proper noun reading, named entity information helps by guiding the search for the term as a proper noun.

The named entity recognition was applied both on English social media texts (tweets) and on an annotated Amharic named entity corpus. Rather than restricting to word space models, we used both a supervised method, namely Conditional Random Fields (CRFs), an evolutionary feature optimization approach (Differential Evolution), and a deep learning approach in the form of a recurrent neural network (bi-directional long short term memory). A large set of different features was developed for identifying and classifying the names and the classifiers were trained using these. The modules and set-ups for Twitter Named Entity Recognition in English tweets are discussed in Chapter 4, while the application for deep learning to Amharic Named Entity Recognition is the topic of Chapter 5.

Chapter 2

Domain Adaptation

Domain Adaptation is a common term for applications that are adapted to a particular text *domain*, e.g., Machine Translation, Named Entity Recognition, Spell Checking or Dialogue Systems. This is a relevant aspect for many of the NLP applications facilitated by the availability of digitized corpora in emerging economies such as Ethiopia.

In the following, some background and definitions of this process will be given. Finally, some experiments will be presented. Common to many of the approaches, domains may arise from a body of text, that is somehow segmented into smaller portions. This entails that experiments extracting semantic relations from general corpora can also be done on domain specific corpora that have been segmented beforehand.

Text corpora, such as the crawled corpora of Amharic, Oromo, Somali and Tigrinya can form the basis of these processes. Using algorithms like Latent Semantic Indexing, Random Indexing, Word2Vec or Self-Organizing Maps, allows for the application to interface with the text resources in order to leverage knowledge therein to achieve Domain Adaptation. Often, the semantic processing can be done off-line prior to running, while it is leveraged in this on-line phase. Thus, different applications can interface with the same resources, e.g., to find related expressions or documents.

2.1 Linguistic Background of Domains

In Linguistics, the understanding of *domains* is tied to Lexical Semantics, i.e., the study of the word *meaning* as opposed to a grammatical function. Braune et al. (2012) argued that errors due to unseen words and word senses account for most errors when moving into a new domain in Machine Translation. Thus, word meaning is an important aspect of Domain Adaptation also in Machine Translation.

The history of Lexical Semantics since 1830 can be divided into three phases, viz. (i) Historical-Philological (ii) Structuralist and (iii) Post-Structuralist Semantics (Geeraerts, 2010). In the second of these phases, Lexical Field Theory (also known as Semantic Fields) (Trier, 1973) emerged as a way of explaining the structure of language and the relations between words.

	Medicine	Computer Science
HIV	1	0
AIDS	1	0
virus	0.5	0.5
laptop	0	1

Table 2.1: Example of a Domain Model.

Trier (1973) characterized such lexical (semantic) fields as mosaics of sets of related lexical items with interdependent meanings. These delineated lexical fields consist of sense-related words. The theory was not uncontroversial, e.g., since the mosaic metaphor is suggesting hard borders between fields exist. Moreover, a relationship can be inferred between this theory and modern ideas like semantic nets.

2.1.1 A Theory of Semantic Domains

Gliozzo and Strapparava (2009) developed a theory of Semantic Domains based on Lexical Field Theory. The hypothesis of *lexical coherence* conjectures that a large part of the lexical concepts in a specific text belongs to the same domain, and constitutes the founding idea behind the theory. In addition, Wittgensteinian Language Games were used as the theoretical basis for delineating these Semantic Domains by inspecting real data in such a way that Semantic Domains arise from words as they are used (i.e., meaning-is-use). Each of the texts from which Semantic Domains are induced comprise Language Games of their own. While the "theoretical void" of delineation in Lexical Field Theory is filled by positing that Semantic Domains are defined by the properties of domain specific corpora, the delineation of *these* texts is not theoretically addressed.

Nonetheless, the theory established a computational model for Semantic Domains and a way of characterizing them. This is done using Domain Models — a tabular expression of the *relative importance* of terms and domains. Gliozzo and Strapparava (2009, p.20) defined Semantic Domains as:

> "Semantics Domains are common areas of human discourse, such as Economics, Politics, Law, Science, etc, which demonstrate lexical coherence."

An example of a Domain Model is found in Table 2.1. The numbers indicate the *Domain Relevance*, for which a function $R(D_z, O)$ is defined to indicate the relevance of a domain D_z to a linguistic object, O. Domain Relevance could be established either from hand-made sources such as WordNet Domains (Magnini and Cavaglià, 2000) or via unsupervised algorithms such as Latent Semantic Indexing.

Gliozzo and Strapparava (2009) applied Domain Models to several NLP tasks, such as Word Sense Disambiguation and (Cross-lingual) Text Categorization, reporting consistent performance improvements. By building Multilingual Domain Models from parallel corpora, com-

- limited subject matter,
- lexical, syntactic and semantic restrictions,
- *deviant* rules of grammar,
- high frequency of certain constructions,
- text structure, and
- use of special symbols.

Figure 2.1: Properties of sublanguages (Lehrberger, 1982).

parable corpora, and dictionaries, such models could be used to ascribe similarity between documents in different languages.

2.1.2 Domains vs. Sub-languages

Domains and Domain Adaptation (primarily in the context of Machine Translation systems) were discussed also before the emergence of the data-driven translation era, then usually as *sublanguages* (Lehrberger, 1982; Luckhardt, 1991). Kittredge (1983) observed:

> “Sublanguages have been characterized in various ways, but there is no widely accepted definition of them.”

Sublanguages were regarded it as restrictions of natural language such that semantic processing was feasible. Moreover, it was stated that these sublanguages are potentially *infinite*. Kittredge considered the following factors necessary restrictions for efficient semantic processing: (i) restricted domain of reference (ii) restricted purpose and orientation (ii) restricted mode of communication and (iv) community of participants sharing specialized knowledge. Evidently, this usage of the term *domain* was in reference to something more specific than the sublanguage itself; rather what the sublanguage refers to.

In studies of translation Lehrberger (1982) presented a descriptive study of a text corpus comprised by instructions for aircraft maintenance of about 70,000 words, regarding — and analyzing — it as a sublanguage. As a result, the properties of sublanguages were summarized, presented in Figure 2.1.

The following property of sublanguages was also observed:

> “It should be clear /.../ that a sublanguage is not simply an arbitrary subset of the set of sentences of a language.“

However, the term *sublanguage* has not been uncontroversial, as the connotations to set theory presupposes that there exists a superlanguage from which sublanguages can be separably cut out (Karlgren, 1993). Karlgren argues that this could make the concept seem unrealistically simple (and prefers the term *register*). However, the disagreement appears largely about the semantics of the term itself, as Lehrberger (1982) states clearly that

> Furthermore, sublanguages overlap and their interrelations form a part of the description of the language as a whole. A language is not simply a union of sublanguages ...

Melby (1997) defined a sublanguage as the following:

> "A sublanguage could be considered to be a case of domain-specific language that is naturally rather than artificially controlled"

thereby distinguishing between the terms.

2.2 Domains and Domain Adaptation

In Mathematics, a *domain* of a function is the set of arguments over which the function is defined. Unfortunately, there is no such commonly agreed on definition of the domain concept in dialogue analysis and linguistics, nor in many other fields. In Psychology, for example, a domain is understood to pertain to the way knowledge is stored in the human mind; however, there is no clear sense of what such domains actually are, but rather that they represent common knowledge, also known under names such as *domain-specific knowledge*, *subject-matter-knowledge*, and *content-specific knowledge* (Alexander, 1992). Carpuat (2014) noted the lack of a definition of what a domain is in the context of machine translation, but related it to topics, genres and registers (sub-languages), while Plank (2011) defined a textual domain as a hypernym over any kind of text variability, such as topic, genre, style medium and vocabulary, and presented empirical evidence on the variability of sentence length across domains as another dimension. In a similar, but more concrete fashion, we define the concept of a dialogue domain as follows:

> **Definition 1. (Linguistic) Dialogue Domain**
> *A* dialogue domain *incorporates the (linguistic)* entities *referred to in a dialogue, as well as the* ways the entities are referred to*, including both the distinct and general vocabulary utilized, and the distinct and general grammatical structures, dialogue acts and strategies used.*

Note that this definition deviates from, e.g., the definition of a domain in the model-theoretic tradition following Montague (Partee, 1976), in that it not only includes the *discourse entities*, but also the *interpretation function* and the *vocabulary* (hence rather the entire model, in the Montagovian sense), and in addition the particular *language structures*, as well as the *dialogue acts* and dialogue strategies.

Definition 1 also deviates from the theory of Semantic Domains developed by Gliozzo and Strapparava (2009) on the basis of Lexical Field Theory (Trier, 1973), and built on the hypothesis of *lexical coherence*, which conjectures that the majority of the lexical concepts in a text belong to the same domain. These Semantic Domains are centered around a Domain Model, which is a tabular expression of the *relative importance* of terms in domains. Then domain relevance can be established either from hand-made sources such as WordNet domains (Magnini and Cavaglià, 2000) or via unsupervised algorithms such as Latent Semantic Indexing (Deerwester et al., 1990), but the approach entails that domains arise from a given unit of text. In contrast, Definition 1 allows for domains to be induced from, e.g., unstructured text corpora, without predefined knowledge of how many domains the texts should be assigned to.

Furthermore, Definition 1 crucially differs from how document specific properties are interpreted in Information Retrieval (IR), in that it includes both the distinct terms used in a domain *and* the general vocabulary. In an IR setting, the general vocabulary terms would be considered to be 'stop words' conveying no distinguishing properties, and commonly removed, while the domain concept here would include these terms in order to establish any domain specific relationships between themselves or between them and the domain specific vocabulary.

Overall, we will here take the machine learning oriented perspective of Jiang (2008) that Domain Adaptation can be seen as a classification task where the distribution of the instances in the training and target domains differ, but where the aim is to use labeled data from one domain to train classifiers for the other domain. The process of adaptation is thus to pick up novel characteristics in the target domain, so that the resulting, adapted model can correctly classify instances belonging to it. Hence, Domain Adaptation presents a fundamental obstacle for statistically learned models, which assume that the data they are to be tested on is distributed identically to the data on which they were trained.

Chapter 3

Language Identification in Code-Switched Text

Social media texts are often fairly informal and conversational, and when produced by bilinguals tend to be written in several different languages simultaneously, in the same way as conversational speech. Often the messages contain text snippets from several languages, that is, showing code-switching. Sometimes the messages even contain code-mixing, where there is a mix of the languages inside a single utterance or even inside a token itself, and it is a challenging task to identify which languages the different words are written in. The recent availability of large social media corpora has thus also made large-scale code-switched resources available for research.

Das and Gambäck (2013) give a comprehensive overview of the work on code-switching until 2015. Notably, Solorio and Liu (2008) trained classifiers to predict code-switching points in Spanish and English, using different learning algorithms and transcriptions of code-switched discourse, while Nguyen and Doğruöz (2013) focused on word-level language identification (in Dutch-Turkish news commentary). Nguyen and Cornips (2016) describe work on analyzing and detecting intra-word code-mixing by first segmenting words into smaller units and later identifying words composed of sequences of subunits associated with different languages in *tweets* (posts on the Twitter social-media site).

In order to address the issues of evaluation and comparison these new corpora entail, we have defined an objective measure of corpus level complexity of code-switched texts (Gambäck and Das, 2014), which will be discussed briefly below (Section 3.1). The code-mixing measure has been developed and tested as described in a series of papers (Das and Gambäck, 2014; Jamatia et al., 2015, 2016), but in particular a paper published at the 10th International Conference on Language Resources and Evaluation (LREC'16) finalizes the measurement and also shows how this formal measure can be used in practice, by applying it to several code-switched corpora (Gambäck and Das, 2016).

A data challenge / shared task on language identification in code-switched text was organized at EMNLP 2014 (Solorio et al., 2014) and repeated in 2016 (Molina et al., 2016), with new datasets. We participated in the shared task and proposed a supervised approach to language identification in code-switched data, framing this as a sequence labeling task where the label of each token is identified using a classifier based on Conditional Random Fields and trained on a range of different features, extracted both from the training data and by using information from Babelnet and Babelfy. The language identification system was published in Proceedings of the Second Workshop on Computational Approaches to Code Switching, EMNLP-2017 (Sikdar and Gambäck, 2016b), with the key parts described here (Section 3.2).

3.1 Measuring Code-Switching in Corpora

When comparing different code-switched corpora to each other, it is desirable to have a measurement of the level of mixing between languages, in particular since error rates for various language processing application would be expected to increase as the level of code-switching increases. Both Kilgarriff (2001) and Pinto et al. (2011) discussed several statistical measures that can be used to compare corpora more objectively, but those measures presume that the corpora are essentially monolingual.

Debole and Sebastiani (2005) analysed the complexity of the different subsets of the Reuters-21578 corpus in terms of the relative hardness of learning classifiers on the subcorpora, a strategy which does not assume monolinguality in the corpora. However, they were only interested in the relative difficulty and give no measure of the complexity as such. In Gambäck and Das (2014) we instead suggested an initial Code-Mixing Index to assess the level of code-switching in an utterance. This measure will be taken as the starting point, and elaborated on here.

3.1.1 Utterance Level Switching

If an utterance x only contains language independent tokens, its code-mixing is zero; for other utterances, the level of mixing depends on the fraction of language dependent tokens that belong to *the matrix language* (the most frequent language in the utterance) and on N, the number of tokens in x except the language independent ones (i.e., all tokens that belong to any language L_i):[1]

$$C_u(x)=\begin{cases} \dfrac{N(x)-\max\limits_{L_i\in\mathbb{L}}\{t_{L_i}\}(x)}{N(x)} & : N(x)>0 \\ 0 & : N(x)=0 \end{cases} \tag{3.1}$$

($L_i \in \mathbb{L}$, the set of all languages in the corpus; $1 \leq \max\{t_{L_i}\} \leq N$). Notably, for mono-lingual utterances $C_u = 0$ (since then $\max\{t_{L_i}\} = N$).[2]

---

[1] Note that the formula in Equation 3.1 differs from, but is equivalent to, the one given in Gambäck and Das (2014).

[2] Consider, e.g., an utterance U_1 with 10 words. If 5 of the words come from language L_1 and the other from language L_2, its C_u will be $(10-5)/10 = 0.50$. However, another 10-word utterance U_2 with all words coming

This initial measure has several short-comings. In particular, it does not reflect what fraction of a corpus' utterances contain code-switching, nor take into account the number of code alternation points: arguably, a higher number of language switches in an utterance increases its complexity, while a corpus with a larger fraction of mixed utterances is (on average) more complex.[3]

Two main sources of information will be utilized to fully account for the code alternation at utterance level: the ratio of tokens belonging to the matrix language ($f_m = [N - max\{t_{L_i}\}]/N$ as in Equation 3.1) and the number of code alternation points per token ($f_p = P/N$, where P is the number of code alternation points; $0 \leq P < N$).

There are many ways to combine two (or several) information sources, in particular if they are independent; see, e.g., Genest and McConway (1990) for an overview. However, P partially depends on $\max\{t_{L_i}\}$,[4] which, for example, rules out the common *logarithmic opinion poll*:

$$(3.2) \qquad p(x) = \prod_{k=1}^{n} p_k(x)^{w_k} \qquad : \sum_k w_k = 1$$

Instead we will use the *linear opinion poll:*

$$(3.3) \qquad p(x) = \sum_{k=1}^{n} w_k \times p_k(x) \qquad : \sum_k w_k = 1$$

Combining $f_m(x)$ and $f_p(x)$ gives a revised utterance level measure for $N(x) > 0$:

$$(3.4) \qquad \begin{aligned} C_u(x) &= w_m f_m(x) + w_p f_p(x) \\ &= w_m \frac{N(x) - \max\limits_{L_i \in \mathbb{L}} \{t_{L_i}\}(x)}{N(x)} \cdot 100 + w_p \frac{P(x)}{N(x)} \cdot 100 \\ &= 100 \cdot \frac{w_m \Big(N(x) - \max\limits_{L_i \in \mathbb{L}} \{t_{L_i}\}(x) \Big) + w_p P(x)}{N(x)} \end{aligned}$$

where w_m and w_p are weights ($w_m + w_p = 1$). Again, $C_u = 0$ for mono-lingual utterances (since then $\max\{t_{L_i}\} = N$ and $P = 0$).

---

from different languages gets $C_u(U_2) = (10 - 1)/10 = 0.90$, correctly reflecting the intuition that U_2 presents a more complex mix.

[3] Compare two 4-word utterances U_3 and U_4 with 2 words each from the languages L_1 and L_2. Thus $C_u(U_3) = C_u(U_4) = (4 - 2)/4 = 0.50$. But if U_3 only contains 1 code alternation point (e.g., if the words are $w_{L_1} w_{L_1} w_{L_2} w_{L_2}$), while U_4 contains 3 switches (e.g., $w_{L_1} w_{L_2} w_{L_1} w_{L_2}$), then U_4 will most likely be more difficult to analyse.

[4] In utterance U_2 with 10 words, all from different languages ($\max\{t_{L_i}\} = 1$), there must be a code alternation between each word, so $P = 9$. If instead $\max\{t_{L_i}\} = 2$, then $4 \leq P \leq 9$, since the utterance, e.g., can contain 2-token sequences from five languages ($P = 4$) or ten 1-token sequences from up to eight different languages.

3.1.2 Corpus Level Switching

Moving to corpus level, the measure could be defined simply as average utterance level switching, as in Equation 3.5

$$C_{avg} = \frac{1}{U}\sum_{x=1}^{U} C_u(x) \tag{3.5}$$

where U is the number of utterances in the corpus.

However, that would ignore two important points: that C_u does not account for code-alternation *between* two utterances, and that the frequency of code-switched utterances in a corpus increases its complexity.

Hence, when combining several utterances, an utterance's matrix language and the matrix language of the previous utterance need to be represented (if they differ, that implies adding a code-alternation point between the two utterances).[5] For each pair of utterances, a factor must be included to account for this, as shown in Equation 3.6:

$$C_u(x{-}1, x) = C_u(x{-}1) + C_u(x) + w_p\delta(x) : \left\{ \begin{array}{rl} L_{x-1} &= \max\limits_{L_i \in \mathbb{L}}\{t_{L_i}(x{-}1)\} \\ L_x &= \max\limits_{L_i \in \mathbb{L}}\{t_{L_i}(x)\} \\ \delta(x) &= \begin{cases} 0 : x{=}1 \vee L_{x-1} = L_x \\ 1 : x{\neq}1 \wedge L_{x-1} \neq L_x \end{cases} \end{array} \right\} \tag{3.6}$$

For combining a corpus' all utterances, we take inspiration from readability indices that are purely word frequency-based and (as C_u), e.g., make no distinction between different word classes. Those are calculated using the average sentence length and another factor, e.g., the average number of syllables per word as in the 'Reading Ease' score (Flesch, 1948), the frequency of multi-syllabic words in 'Fog' (Gunning, 1952), or the frequency of long words in 'LIX' (Björnsson, 1968).

Flesch' Reading Ease score is based on the average number of words per sentence and average number of syllables per word:

$$\mathrm{RE} = 206.835 - [1.015 \cdot (\frac{\mathrm{W}}{\mathrm{S}}) + 84.6 \cdot (\frac{\mathrm{L}}{\mathrm{W}})] \tag{3.7}$$

where W is the number of words in the text, S the total number of sentences, and L the total number of syllables (hence words/sentence are weighted as $1.2 \cdot$ syllables/word).

---

[5] This is different from, and more important than, checking whether the language of an utterance's first token differs from that of the previous utterance's last token.

The Fog Index is the number of words per sentence plus the percentage of multi-syllabic words:

$$\text{Fog} = 0.4 \cdot [\frac{\text{W}}{\text{S}} + 100 \cdot (\frac{\text{F}}{\text{W}})] \tag{3.8}$$

where W is the number of words in the text, S the number of sentences, and F the number of "foggy" words, that is, mainly words with more than three syllables.

The LIX measurement is the number of words per sentence plus the percentage of long words:

$$\text{LIX} = \frac{\text{W}}{\text{S}} + 100 \cdot (\frac{\text{L}}{\text{W}}) \tag{3.9}$$

where W is the number of words in the text, S the number of sentences, and L the number of long words (defined as words with more than five characters).

In the case of code-switching, the first factor is the average switching level per utterance, as calculated by inserting the C_u given by Equation 3.6 into the average of Equation 3.5, while the second factor is the frequency of utterances that contain any code-switching (i.e., utterances with $C_u > 0$). Thus arriving at Equation 3.10:

$$\begin{aligned} C_c &= \frac{\sum\limits_{x=1}^{U} C_u(x) + w_p\delta(x)}{U} + w_s\frac{S}{U} \cdot 100 \\ &= \frac{100}{U}\left[\sum_{x=1}^{U}\Big(w_m f_m(x) + w_p\left[f_p(x) + \delta(x)\right]\Big) + w_s S\right] \end{aligned} \tag{3.10}$$

where S is the number of utterances that contain code-switching ($0 \leq S \leq U$), and w_s the relative weight attached to the switching frequency.

3.1.3 Comparing Corpora Level Switching

The main issue when applying an information source combination method (e.g., Equation 3.2 or 3.3) is how to choose the weights, and several strategies have been proposed. We tried a number of them experimentally at the utterance level, but the only combination giving reliable and intuitive values was the average (equal weights: $w_k = \frac{1}{n}$) reflecting an observation also made by Clemen (2008): "*Having spent much of my career studying various combination methods, it has been somewhat frustrating to consistently find that the simple average performs so well empirically.*" (p.765)

With two information sources, the weights are $w_m = w_p = \frac{1}{2}$ and Equation 3.4 (for $N > 0$) reads:

$$C_u(x) = 100 \cdot \frac{N(x) - \max\limits_{L_i \in \mathbb{L}}\{t_{L_i}\}(x) + P(x)}{2N(x)} \tag{3.11}$$

Similarly, for determing the relative weight attached to the switching frequency at the corpus level (w_f in Equation 3.10), we again compare to readability indices. Fog and LIX combine two information sources without weighting (i.e., indirectly use the average); however, Reading Ease applies a weighting which treats the {words/sentence} factor as 1.2 × {syllables/word}. Flesch (1948) derived this through regression based on correlations between the average grade of children and those who could answer 50% and 75% of some test questions.

Using these weights, Equation 3.10 becomes

$$\begin{aligned} C_c &= \frac{100}{U}\left[\frac{1}{2}\sum_{x=1}^{U}\Big(f_m(x)+f_p(x)+\delta(x)\Big)+\frac{5}{6}S\right] \\ &= \frac{100}{U}\left[\frac{1}{2}\sum_{x=1}^{U}\Big(1-\frac{\max\limits_{L_i\in\mathbb{L}}\{t_{L_i}\}(x)-P(x)}{N(x)}+\delta(x)\Big)+\frac{5}{6}S\right] \end{aligned} \tag{3.12}$$

This objective measure of the complexity of code-switched texts has been tested on a range of social media corpora written in a variety of combinations of languages. Certainly, though, no such measure will ever be able to capture all types of differences between corpora. In particular, the ways corpora were collected and their intended usage also need to be taken into account. However, levelling out such differences should arguably not be the aim of the code-switching measure itself, but rather be left to the users: when comparing corpora with widely different scopes, the users themselves need to be aware of the potential variation and consider this when deciding on whether a straight-forward comparison really makes sense.

The next section will go into one of the key problem areas related to code-switching, namely the identification of which languages a particular text is a mix of, and most vitaly, which language every word in those texts belong to, that is, the word-level language identification and disambiguation task for code-switched texts.

3.2 Language Identification

A first code-switching data challenge was organized at EMNLP 2014 (Solorio et al., 2014). The task was to identify the language for each word in a text, classifying the words according to six labels: ‘Lang1’, ‘Lang2’, ‘Mixed’, ‘NE’, ‘Other’, and ‘Ambiguous’. The first two labels identify tokens from the main languages that are mixed in the text, while the third is for tokens with word-internal mixing between these languages; ‘NE’ for named entities; ‘Other’ for language independent tokens (punctuation, numbers, etc.) and tokens from other languages, and ‘Ambiguous’ denotes tokens that cannot safely be assigned any (or only one) of the other labels. This shared task was organized again in 2016 (Molina et al., 2016), with new datasets and slightly different labels, adding ‘Unk’ for unknown tokens.[6]

We participated in the shared task and proposed a supervised approach to language identification in code-switched data, framing this as a sequence labeling task where the label of each token is identified using a classifier based on Conditional Random Fields (CRF) and trained on

---

[6] An eighth label ‘FW’ was included for foreign words, but no words in the English-Spanish corpora were tagged with it.

a range of different features, extracted both from the training data and by using information from Babelnet and Babelfy. We used the C^{++}-based CRF^{++} package (Kudo, 2013), a simple, customizable, and open source implementation of Conditional Random Fields for segmenting or labelling sequential data. Conditional Random Fields (Lafferty et al., 2001a) are conditional, undirected graphical models that can easily incorporate a large number of arbitrary, non-independent features while still having efficient procedures for non-greedy finite-state inference and training.

The CRF classifier was trained and tested on the datasets annotated by the shared task organizers (Molina et al., 2016).[7] Three types of data were provided: training, development and test, all containing tweets written in either English, Spanish or a mix of the two languages. In the training and development datasets, the total number of tweets are 11,400 and 3,014, respectively, with language identification offsets given for each category. In the test data, the total number of tweets is 17,723 without annotations. 7,724 of the tweets in the test data actually contain some code-switching, while the remaining 9,999 are monolingual. If tweets that contain URLs are disregarded, there are 10,716 tweets in the test data, of which 4,626 contain code-switching (and hence 6,090 that do not contain any language mixing).

3.2.1 Features based on training data

Two sets of features were developed to train the CRF-based language identification model: one extracted from the training data and the other based on information from Babelnet (Navigli and Ponzetto, 2012) and Babelfy (Moro et al., 2014), with most of the features and their settings being based on the training data. The complete set of features induced from training data was as follows:

Local context. Local contexts play an important role for identifying the languages. Here the two preceding and two succeeding words were used as local context.

Word suffix and prefix. Fixed length characters stripped from the beginning and ending of the current word. Up to 4 characters were removed.

Word length. Analysis of the training data showed that the Spanish words on average were shorter than the English words. Words with 1–4 characters were flagged with a binary feature.

Word previously occurred. A binary feature which checks if a word already occurred in the training data or not (i.e., if the word presents in the training data, the feature is set to 1 otherwise 0).

Initial capital. In general, proper nouns tend to start with capital letters, so this feature checks whether the current word has an initial capital.

All capitals. A binary feature which is set if the current word contains only capital letters. The feature is very helpful for identifying named entities (since, e.g., abbreviations often refer to named entities).

---

[7] `care4lang1.seas.gwu.edu/cs2/call.html`

Single capital letter: checks if the word contains a single capital letter or not.

All digits: set to 1 if the word contains only numerical characters. This is helpful for identifying tokens belonging to the 'Other' category.

Alphanumeric: a binary feature which flags if the word contains only digits and alphabetical characters together.

All English alphabet: checks if all a word's characters belong to the English alphabet.

Special Spanish character: a flag which is set if the current word contains any Spanish-specific letters (á, é, etc.).

Hash symbol: set to 1 if a word contains the symbol '#', otherwise 0.

Rate symbol: set to 1 if the current word contains the symbol '@'.

Word with single letter. Many single letter words were observed to belong to Spanish, so this flag is set if the word length is exactly 1.

Two consecutive letters. Some words repeat two character sequences several times (e.g., *hahaha*, *jaja*). Each token is split into two character sequences and this binary feature is set if each two letter character sequences matches.

Same letter occurred multiple times. Many words in tweets contain sequences of the same character repeated many times (e.g., ewww, yaaaas). The feature is set if a letter occurred in a word more than two times consecutively.

Gazetteer NE list. A list of named entities (NE) was collected from the training data. This flag is set if a token matches an item on the NE list.

Special character list. A list of special characters (e.g., emojis) was collected from the training data. If a tokens contains any character which is on the list, the binary feature is set.

3.2.2 Babelnet and Babelfy features

Three further features were developed from external resource, Babelnet (Navigli and Ponzetto, 2012) and Babelfy (Moro et al., 2014):

WordNet feature: Every token is passed to the Babelnet database for checking whether the token exists in the English WordNet or not. If the token appears in the database, the feature is set to 1, otherwise to 0.

Multilingual WordNet: The Babelnet Multilingual WordNet is checked for Spanish, by passing each token to the Babelnet database and checking whether the token is present in the database or not.

Babelfy Named Entity: Named entities are extracted from Babelfy and used as a feature, which is utilized for identification of the 'NE' category tokens.

System setup	**Mono-lingual**			**Code-switched**			**Weighted**	**Token-level**
	P	R	F_1	P	R	F_1	F_1	**Accuracy**
Without external resources	0.94	0.92	0.93	0.82	0.85	0.83	0.904	0.943
With external resources	0.95	0.93	0.94	0.82	0.87	0.85	0.911	0.952

Table 3.1: System performance on the development data, with and without the external resources (Babelnet and Babelfy)

3.2.3 Results

There were two types of evaluation, at tweet level and at token level. On the test data, performance was checked on tweets with and without URLs. The tweet level precision (P), recall (R) and F_1-scores obtained on the monolingual part of the development data were 95%, 93% and 94%, respectively. On the code-switched part, the precision, recall and F_1-scores were 82%, 87% and 85%, giving a weighted, total F_1-score of 91.1%. At token level, the development data accuracy was 95.2%, as shown in Table 3.1.

Table 3.1 also gives the development data scores for a system trained without the second feature set, i.e., without the Babelnet and Babelfy features. As can be seen in the table, the contribution from those features is small but useful, adding 0.9% to the token-level accuracy and 0.7% to the tweet-level weighted F_1 score, with the main contribution (2%) being on recall for the tweets containing code-switching.

Applying our system to the test data excluding tweets with URLs, the tweet level monolingual, code-switched and weighted F_1-scores were 91%, 87% and 89%, with a token level accuracy performance of 96.5%. When including all tweets, the same tweet level F_1-scores were 90%, 85% and 87.4%, and the token level accuracy 95.7%.

Overall nine teams participated in the shared task, with the top-6 teams obtaining fairly similar performance on all measures, and both at token and tweet level, although some systems performed clearly better when the tweets containing URLs were excluded. For both evaluation levels, our system achieved fifth place in the shared task when excluding tweets with URLs, and third (token) and fourth (tweet) place when including all tweets, with a token level accuracy difference between our system and the best performing systems of only 0.8% and 0.4%, respectively. At the tweet level, the system performed relatively better on monolingual tweets than on tweets that contained some code-switching, when compared to the other top systems.

Chapter 4

Twitter Named Entity Recognition

Named entity recognition is the process of identifying proper names and classifying them into some predefined labels/categories. Named Enties play an important role to help the semantic search in a more efficient way, for example, by identifying whether the search term 'apple' refers to an edible object or a company. The named entity recognition was applied on social media texts (tweets). Systems that simultaneously identify and classify named entities in Twitter typically show poor recall. To remedy this, the task is here divided into two parts: i) Named entity extraction from tweets and ii) Twitter name classification into ten different types.

The noisiness of the texts makes Twitter named entity (NE) extraction a challenging task, but several approaches have been tried: Li et al. (2012) introduced an unsupervised strategy based on dynamic programming; Liu et al. (2011) proposed a semi-supervised framework using a k-Nearest Neighbors (kNN) approach to label the Twitter names and gave these labels as an input feature to a Conditional Random Fields, CRF (Lafferty et al., 2001a) classifier, achieving almost 80% accuracy on their own annotated data. Supervised models have been applied by several authors, e.g., Ritter et al. (2011) who applied Labeled LDA (Ramage et al., 2009) to recognise possible types of the Twitter names, and also showed that part-of-speech and chunk information are important components in Twitter NE identification.

First, we used a supervised method, namely Conditional Random Fields (CRFs) to identify and classify Twitter names. A large set of different features was developed for identifying and classifying the named entities, and the CRF-based classifier was trained using these. The features can be divided into three sets: i) Lexicon-based features, ii) Character-based features and iii) Context-based features. The feature-rich Twitter Named Entity Recognition and Classification system was published in the Noisy User-generated Text (W-NUT) shared task at COLING-2017 (Sikdar and Gambäck, 2016a). For Twitter named entity recognition on unseen test data, our system obtained the second highest F_1 score in the shared task: 63.22%. The system performance on the classification task was worse, with an F_1 measure of 40.06% on unseen test data, which was the fourth best of the ten systems participating in the shared task.

As an alternative and in order to improve the Twitter named identification accuracy, another Twitter named entity identifier was developed using a CRF-based model together with an unsupervised method, multi-objective framework built on Differential Evolution (DE). Differential Evolution is an evolutionary algorithm, which not only optimises the feature set by selecting the most relevant features, but also identifies the proper context window for each selected feature. The named entity classification was done using Vector Space Modeling and edit distance techniques. Several different Twitter named entity models were generated and later the outputs of the different models were combined to enhance the system accuracy. The approach obtains F-scores of 70.7% for Twitter named entity extraction and 66.0% when subsequentially aiming to link the extracted entities to the DBpedia database. This work was published in ICON 2016: the 13th International Conference on Natural Language Processing (Sikdar and Gambäck, 2016c).

4.1 Features for Twitter Named Entity Recognition

A range of different features were developed for identifying the Twitter names. These features are described below. Note that some of the feature types are implemented as several different features and that the first five types mainly are lexicon-based, while the last five types primarily relate to the context. The eight feature types in the middle of the list are predominantly word internal and character-based. The contributions of these three different groups of feature types will be compared to each other in the next section.

Lexicon-based features

Lexical data: This binary feature was extracted from the lexical data supplied by the W-NUT shared task organisers. The feature is set to 1 if the current word belongs to the lexical data, otherwise 0. In the DE-based experiments, this feature was based on a Gazetteer list automatically extracted from the training data.

Babelfy named entities: Each tweet was passed to the Babelfy (Moro et al., 2014) named entity recognition system for recognizing Twitter names. If the current word belongs to the Babelfy named entities, this binary feature is set. No Babelfy information was used in the DE experiments, but only a flag for if the word had been seen previously in the data.

Part-of-speech (POS): The TweeboParser[1] was used for generating part-of-speech tags for each token in the tweets and the POS tag of the current word was used as a feature. The POS tags of the previous two tokens and following two tokens were also used as features, so in total there are five POS tag features for each token.

Stop word match: All tokens are checked against a stop word list collected from the web.[2] The binary feature is set if the current token matches one of these stop words.

Word frequency: Less frequent words were found to often belong to named entities. If the pre-calculated frequency from the training data of the current word is less than a certain threshold, this binary feature is set.

Character-based features

Numeric: Many Twitter names were found to contain numeric characters, so this binary feature shows whether the current token contains numeric characters or not. In the

---

[1] `http://www.cs.cmu.edu/~ark/TweetNLP/`

[2] `http://www.ranks.nl/stopwords`

DE-based experiments, this feature was split into two: one for at least one numeric character and one for tokens with only digits.

Initial capital: Proper nouns in general tend to start with capital letters, so this feature flags if the current word has an initial capital letter.

Inner capital: This feature is set if the token contains any capital letter in a word-internal position. In the DE-based experiments, two features were added for tokens with only one capital and for those with only capitals.

Word normalisation: The word mapped to its equivalent class: each capital letter in the token is mapped to 'A', all lower-case letters to 'a', and digits to '0'. Other characters are kept unaltered.

Special character followed by token: Many Twitter names follow certain special characters (e.g., '@' and '#'). The feature checks if the token is following any special character or not.

Word length: If the current word's length is greater than some threshold (e.g., ≥ 5), this binary feature is set.

Word suffix and prefix: A fixed maximum number of characters (e.g., 4) are stripped from the beginning and the end of the current word, with the remainder parts being used as two features.

Context-based features

Local context: Local contexts play an important role in identifying Twitter names. In the W-NUT experiments, we used the three previous and the three next words as local contexts (so there are six context features). In the DE experiments, only the preceding and suceeding words were used as local context.

Chunk information: Chunk information was collected using annotated chunk data from the OSU Twitter NLP Tools.[3] A CRF-based chunk model was developed using prefix, suffix and part-of-speech information. Each tweet was passed to the chunk model and chunk labels generated for that tweet. This information was then used as a feature for the current token. In addition, the chunk information of the previous token and the following token were also used as features. (The chunk information was not utilized in the DE-based experimements.)

First word: This binary feature checks if the current token is at the beginning of a sentence or not.

Last word: This binary feature is set if the current token is the last word of a sentence, without considering sentence ending symbols and punctuation marks (i.e., '.', '?', '!', etc.).

Previous label: Finally, the previous token's calculated named entity label was used as a feature.

For the actual named entity system, some of the limits used for the features were set based on testing on the training data. Hence, the thresholds for the word frequency and word length features were empirically set to ≥ 10 and ≥ 5, respectively, while the prefix and suffix strip lengths were fixed to four characters.

---

[3] `https://github.com/aritter/twitter_nlp/blob/master/data/annotated/chunk.txt`

Features	**Recall**	**Precision**	**F$_1$**
Lexical	57.19	49.48	53.05
Word internal	44.18	49.32	46.61
Context	16.79	57.81	26.03

Table 4.1: Feature class contributions to Twitter Named Entity Identification ('notypes' development data)

4.2 Supervised Named Entity Identification and Classification

The Twitter Named Entity Recognition shared task (Strauss et al., 2016) at W-NUT 2016, the COLING workshop on noisy user-generated text, was divided into two subtasks: 'notypes' and '10types'. In the 'notypes' subtask, named entities should just be identified in the tweets, while in the '10types' subtask the aim was to identify and classify Twitter names into ten different categories: *facility, geo-loc, movie, musicartist, person, company, product, sportsteam, tvshow*, and *other*. We hence first identify the Twitter names from the tweets, and then in a second step classify the names according to the ten given labels.

The Twitter named entities were first extracted from the tweets using a supervised machine learning approach, namely Conditional Random Fields, CRFs (Lafferty et al., 2001a). We used the C^{++} based CRF^{++} package,[4] a simple, customizable, and open source implementation of CRF for segmenting or labelling sequential data. For the 'notypes' task, the system was developed on the training data and evaluated on the development data. The 'notypes' model was initially developed using all the training data provided by the shared task organizers (Strauss et al., 2016), but that gave low recall in relation to the precision. To increase the recall performance, the tweets that did not contain any named entities were removed from the training data, and a new model was built using all the features described in Section 4.1. This system achieved recall, precision and F-measure values of 68.68%, 65.14% and 66.86%, respectively. This F-measure value represents an increase by almost 7% compared to the model which was built using all the training data.

Table 4.1 looks at the feasibility of the different groups of features, that is, how much each feature group actually contribute in isolation. In the table, the features are sorted into three different classes: 'lexical', 'word internal' and 'context'. Referring to Section 4.1, the 'lexical' group consists of the features lexical data, babelfy, POS, stopword and word frequency. The 'word internal' (character-based) feature group contains the alphanumeric, initial/inner capital, normalisation, special character, word length and pre-/suffix features, while the 'context' group is made up of the context window words features, chunk, first/last words, and the previous token's label. As can be seen in Table 4.1, the context features do not contribute much at all to improving the recall, but are the most helpful features for improving the precision. The lexical and word internal features contribute roughly equally to the precision score, but the lexical features very clearly are the most useful for achieving good recall.

---

[4] `http://crfpp.sourceforge.net`

The full-feature set ‘notypes’ model built only on the tweets with named entities was entered in the W-NUT shared task and when applied to the unseen test data it achieved state-of-the-art results by obtaining the second highest score in the task. The system achieved recall, precision and F_1 values of respectively 64.18%, 62.28% and 63.22%, which was 2 points less on F-measure than the best performing system, but 3 points more than the third ranked system.

For the ‘10types’ subtask, the goal was to identify and classify Twitter names into ten given categories. Two models were developed for the Twitter name classification, as follows:

Model-1: In Model-1, the tokens are classified into ten categories using CRF. The same features were utilised as described in the experiments above.

Model-2: This model was based on the lexical data given by the shared task organisers (Strauss et al., 2016): the Twitter named entity classes are categorized into 10types based on the supplied file ‘`dictionaries.conf`’. In addition, Twitter names were extracted from the training data and merged with the lexical data for each category. All tokens were passed to the lexical data and classified into the ten categories.

The output of these two models was first merged and the merged output was later checked as to whether it belonged to ‘notypes’ or not. When merging the output from the models, highest priority was given to Model-2, if the two models generated different named entity classes for a particular token. For example, ‘Washington Navy Yard’ is recognized as belonging to the *other* category by the Model-1, but Model-2 recognizes it as *facility.* So the category actually assigned to this entity by the overall system will be *facility*, as suggested by Model-2. If an entity matches more than one class in Model-2, we randomly assign the class of the entity among the matched classes.

Next, the merged output was compared to the ‘notypes’ entities. If the merged output belongs to those ‘notypes’ entities (fully matched), the output entity is considered to be a Twitter name, otherwise it is discarded. For example, the entity ‘Nobu Restaurant’ is classified as *other*, but this entity is not identified by ‘notypes’, so it is not considered as a Twitter named entity. When using all the feature classes (lexical, word internal and context), Model-1 produced the best recall (28.74%) with an F-measure of 37.40%. Model-2 generated a better precision (85.16%), but due to bad recall still had a lower F_1: 27.63%. Combining the two models boosted the results on the development data on all measures, giving a 43.81% F-score.

Table 4.2 indicates the contribution from each of the three types of feature sets to the performance of Model-1. Just as for the identification task, it is also for classification clear that the context features are most helpful for boosting precision, while the lexical features help the recall most.

Applying the ‘10types’ model to the previously unseen test data, it achieved recall, precision and F-measure values of 53.19%, 32.13% and 40.06%, respectively. The system came in fourth place in this subtask. One of the main reasons why our system is outperformed by the top systems is that not all the named entities identified by the system were classified into any of the ten categories.

Features	**Recall**	**Precision**	**F_1**
Lexical	22.39	34.58	27.18
Word internal	15.13	49.50	23.17
Context	6.20	95.35	11.65

Table 4.2: Feature contributions to Model-1 for Named Entity Classification ('10types' development data)

4.3 Unsupervised Named Entity Extraction and Linking

A specific shared task on Twitter named entity recognition and linking (NEEL) to DBpedia was held at the #Microposts2016 workshop (Rizzo et al., 2016), with the problem defined as to identify named entities from the tweets (called 'Strong_typed_mention_match') and to link them to the DBpedia database ('Strong_link_match'). DBpedia extracts structured information from Wikipedia and links different Web data sets to Wikipedia data, allowing for sophisticated queries against Wikipedia. The DBpedia knowledgebase is available in 125 languages, with the English version describing 4.58 million items, out of which 4.22 million are classified in a consistent ontology.

Five teams participated in the #Microposts2016 NEEL challenge. However, most of the systems suffered from very low recall values in the Twitter NE identification task and were actually unable to efficiently recognise Twitter names: To increase recall and F-score, we take a two-step approach to identifying and classifying named entities in noisy user-generated texts. In the first step, Twitter names are identified using CRF within the framework of Differential Evolution (Storn and Price, 1997). In step two, the named entities are classified into seven categories and linked to DBpedia using a vector space model and edit distance techniques. The identified named entities are also classified using CRF, and the outputs of the classification models are later combined.

To fairly evaluate the approach, it was applied to the unseen test data using the selected features along with the context features, giving the recall, precision and F_1-scores of 73.9%, 89.2% and 80.8%, respectively on the named entity identification. The F-measure performance constitutes an increase by 20 points over a baseline using only a CRF-based classifier. When merging the development data with the training data and building a model using selected features, the recall, precision and F-measure values are 81.3%, 90.8% and 85.7%, respectively, also improving on the scores obtained by the baseline model. The results show that the multi-objective DE-based approach efficiently identifies Twitter names from noisy text.

For the named entity classification, the best model achieved precision, recall and F_1-scores of 65.3%, 67.1% and 66.2%, respectively, on unseen test data, which is by far the the best results compared to all the other models as well as to all the #Microposts2016 systems, with the best existing system, KEA (Waitelonis and Sack, 2016), reaching a 47.2% F-score on the task. Still, when merging the development data with the training data and training on the combined data set, the performance of the DE-based system is even further improved, with precision, recall and F-score of 69.7%, 71.7% and 70.7%.

To link the classified Twitter names to DBpedia, a knowledge-based approach was used (utilizing either training data or the DBpedia database). The DE-based models then outperform the NEEL 2016 systems in terms of precision, but fail get a higher F-score than the KEA system due to lower recall. However, when development data are merged with the training data for the model building, the DE-based approach again gives the best test data results among all the systems, achieving recall, precision and F-measure values of 57.2%, 78.1% and 66.0%, compared to an F-score of 50.1% for KEA.

Chapter 5

Named Entity Recognition for Amharic Using Deep Learning

This chapter will focus on named entity recognition (NER) for Amharic, the second largest Semitic language in the World (after Arabic) and the main language for country-wide communication in Ethiopia. Amharic is written in its own unique script which lacks capitalization and in total has 275 characters (mainly consonant-vowel pairs).

The task of named entity recognition is to identify proper names and classify them into some predefined categories. Most current NER research utilises machine learning approaches (Lafferty et al., 2001b; Takeuchi and Collier, 2002; Zhou and Su, 2002), heavily based on the availability of resources on the web, a route which obviously cannot always be taken for resource-poor languages.

Recently, deep neural networks have been shown to effectively solve several language processing tasks such as part-of-speech tagging, sentiment analysis, and NER. A feed-forward neural network to identify named entities was designed by Collobert et al. (2011) using a fixed number of context words, while Chiu and Nichols (2016) further improved performance using character and word embeddings. Huang et al. (2015) proposed a more complex method based on a bi-directional long short term memory (LSTM) model.

Most named entity recognition efforts have focused on a few European and Asian languages, while African languages have been given little attention; however, three Master's Thesis projects at Addis Ababa University have experimented with subsets of the WALTA corpus (Demeke and Getachew, 2006; Gambäck, 2012) using the version of the corpus transcribed into the Roman alphabet and annotated it with 4-class named entity (NE) tags for persons, organisations, locations and others (non-NE; roughly 90% of the datasets):

Mehamed (2010) and Alemu (2013) both used Conditional Random Fields (CRFs) classifiers trained on different word and context features (word prefixes and suffixes, and the NE and part-of-speech tags of the word), with Mehamed (2010) achieving recall, precision and F_1-measure values of 75.0% , 74.2%, and 74.6%, respectively, for this task on a 10,500 word subset.

Alemu (2013) experimented with context windows of up to two words before and after the current token on another 13,500 word subset of the WALTA corpus, improving recall to 84.9% and precision to 76.8% for an 80.7% F-score. It is not clear from that work how the dataset was selected, or whether it overlapped with the dataset used by Mehamed (2010).

Belay (2014) used a combination of decision trees, support vector machines and hand-crafted rules on the same dataset as Alemu (2013), but artificially expanded it to get a better balance between the classes. However, Belay (2014) noted that his hybrid learning approach was outperformed by a straight-forward baseline method using only part-of-speech information and a binary flag for nominals.

In contrast, we have experimented with a substantially larger dataset and two deep learning approaches to identify and classify Amharic named entities into six predefined classes: Person, Location, Organization, Time, Title, and Other (non-named entity tokens). Both set-ups utilize a recurrent neural network, a bi-directional long short term memory (LSTM) model with various features. Word vectors based on semantic information are built for all tokens using an unsupervised learning algorithm, word2vec. In a basic system set-up, the word vectors were merged with a set of specifically developed language independent features and together fed to the neural network model to predict the classes of the words. When evaluated by 10-fold cross-validation, this basic deep-learning based Amharic named entity recogniser achieved reasonable average precision (77.2%), but did worse on recall (63.4%), for a 69.7% F1-score.

In order to improve performance, a stacked system approach was taken, where a supervised Conditional Random Fields (CRF) classifier trained on language independent features first predicts each token's named entity class. The CRF output predictions, the selected feature set and the word2vec word vectors are then fed to the deep neural network (LSTM model) which assigns labels to the words. 10-fold cross-validation showed the Amharic named entity recogniser producing much improved and very good precision (86.0%), but still doing worse on recall (65.5%), with a 74.3% F-score.

5.1 Amharic Named Entity Annotation

The Polyglot project (Al-Rfou et al., 2013) has created word embeddings for the more than 100 languages that have at least 10,000 Wikipedia entries, which mainly include European and Asian languages (and some artificial languages), but in addition to Arabic also a few Sub-Saharan African languages such as Yoruba, Swahili, Africaans and Amharic.[1] However, although their 4-class annotations (persons, organisations, locations and others) for named entity recognition (Al-Rfou et al., 2015) include 40 languages, Arabic is the only language spoken in Africa which has been annotated so far.[2]

In contrast, the Amharic dataset annotated within the SAY project at New Mexico State University's Computing Research Laboratory uses a richer 6-class annotation scheme, with the categories person, location, organization, time, title and other (for non-named entity tokens).

---

[1] http://bit.ly/embeddings

[2] https://bit.ly/polyglot-ner

Fold	sent.	tokens	NEs
1	3,784	99,095	5,056
2	3,802	98,293	4,894
3	3,859	99,379	4,828
4	3,743	96,282	4,878
5	3,895	100,197	5,018
6	3,862	100,963	5,027
7	3,902	100,877	5,012
8	3,730	96,620	4,788
9	3,832	98,300	4,951
10	3,725	97,078	4,868

Table 5.1: Training data statistics

			Named entities		
Fold	sent.	tokens	total	match	noMatch
1	453	10,581	424	236	188
2	435	11,383	586	249	337
3	378	10,297	652	313	339
4	494	13,394	602	291	311
5	342	9,479	462	200	262
6	375	8,713	453	209	244
7	335	8,799	468	218	250
8	507	13,056	692	275	417
9	405	11,376	529	215	314
10	512	12,598	612	307	305
Total	4,236	109,676	5,480		

Table 5.2: Test data statistics

The SAY annotations are available in 322 XML files from the Lexical Data Repository of the Ge'ez Frontier Foundation.[3]

To run the experiments reported on below, the named entity annotated SAY dataset was here split into ten parts for cross validation. Table 5.1 shows the statistics of the training data in terms of number of sentences, tokens, and named entities. Table 5.2 gives the same information for the test data, including the total number of named entities, as well as the number of test NEs that matched with training NEs ('match') and those that did not ('noMatch').

5.2 A Basic Named Entity Recognition System for Amharic

The task of Amharic named entity recognition system is to identity proper names and classify them into some predefined categories/labels. A deep learning approach was used to recognize

---

[3] `https://github.com/geezorg/data/tree/master/amharic/tagged/nmsu-say`

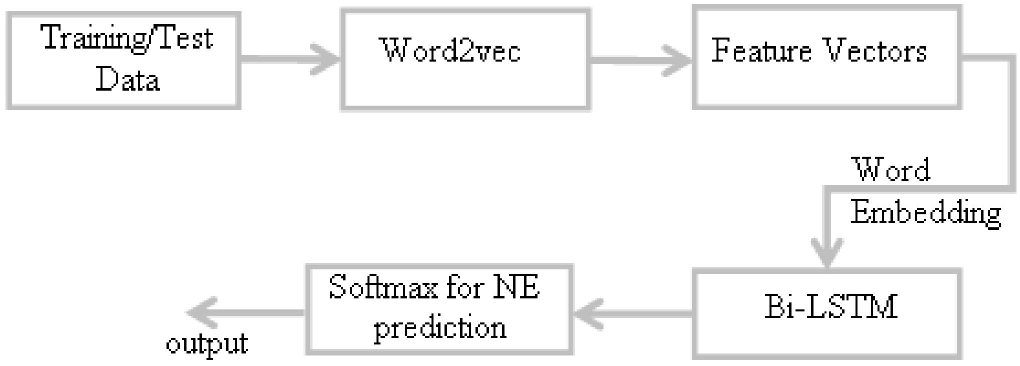

Figure 5.1: Pipeline of the basic LSTM model

the named entities from Amharic text. The pipeline of NE prediction model is shown in Figure 5.1.

5.2.1 Feature Vectors

First a set of language independent features was defined, as follows:

Local context plays an important role in identifying names. Here two words before and after the focus word were used as local context (so there are four context features).

Part-of-speech tags extracted for each token using HornMorpho (Gasser, 2011), which is one of the few resources that already are available for Ethiopian languages and provides some morphological processing for Amharic, Tigrinya and Afaan Oromo.

Word suffix and prefix obtained by stripping a fixed number (up to 4) of characters from the beginning and end of the current word.

Word frequency: Less frequent words were found to often belong to named entities. If the pre-calculated frequency from the training/test data of the current word is less than a certain threshold, this binary feature is set. The thresholds were empirically set to 10 and 4 for training and test data, respectively.

Digit check: to identify the ‘time’ class, it is helpful to mark if a token contains any digit(s). Hence this binary flag is set for tokens containing at least one digit.

5.2.2 Word embeddings

The deep learning method is divided into two parts: word embedding and bi-directional LSTM. The word embeddings are generated in two ways, through word2vec and random vectors. In random vectors, all the words in the corpora are initialized with random values. These random vectors are fed to the bi-directional LSTM model to classify the words into the six categories.

The idea of word2vec method is that words occurring in similar contexts have related meanings. It takes inputs from large corpora and generates a word vector for each word. There are two types of embeddings: continuous-bags-of-words (CBOW) and skip-gram models (Mikolov et al., 2013,?). In the CBOW architecture, the model predicts the current word from a window of surrounding context words. In the skip-gram model, it predicts the context words using the current word. The word2vec model can be trained using a softmax function (Rong, 2014) or negative sampling (Rong, 2014). For Amharic, the entire corpus was used as a training data for a word2vec model using skip-grams and negative sampling. Then a bi-directional LSTM model was utilised to classify the words based on word vectors generated either by word2vec or randomly.

5.2.3 Bi-directional Long Short Term Memory Model

In the bi-directional LSTM model, the network is learned by using training data and tested on unseen data. The network uses an embedding/input layer with two hidden layers. In the output layer, the softmax (Rong, 2014) function classifies the words into six categories/labels. Three different LSTM models were generated using word embedding layers with various features, as follows:

Model-1 uses randomly initialize word vectors to train the LSTM network which classifies the words based on the softmax function.

Model-2 uses word2vec to generate a word vector for each unique word and train the LSTM network classifier on these word vectors.

Model-3 takes as input word embeddings developed using word2vec along with the language independent features (suffix and prefix, POS, frequency and digit-check). For the suffix and prefix features, 5-dimensional word vectors are generated for each length of suffix/prefix character(s) using word2vec. The suffix and prefix lengths are set for up to four characters, so that 40 (5x8) word vectors are generated for the suffix and prefix features. In addition, one-hot vectors are generated for each of the other feature: a length 5 one-hot vector for POS, a length 2 one-hot vector for frequency, and a length 2 one-hot vector for the digit-check feature.

The LSTM network is learned using these word embeddings and classified the words into the six categories, following the NE prediction model pipeline shown above (Figure 5.1).

5.2.4 Results

As outlined in the previous section, three models were built based on a bi-directional LSTM to predict the named entity class of a particular token. Each model was tested using 10-fold cross-validation on the data described in Section 5.1. In the first model (Model-1), the bi-directional

Fold	Precision	Recall	F-measure
1	0.8271	0.3517	0.4935
2	0.8023	0.3436	0.4812
3	0.8788	0.2895	0.4356
4	0.8511	0.2846	0.4266
5	0.7621	0.3173	0.4481
6	0.8540	0.2588	0.3973
7	0.8621	0.3125	0.4587
8	0.8773	0.3187	0.4676
9	0.8101	0.2812	0.4175
10	0.8514	0.3014	0.4452
Avg.	0.8376	0.3059	0.4471

Table 5.3: LSTM random vectors (Model-1)

Fold	Precision	Recall	F-measure
1	0.8067	0.6446	0.7166
2	0.6845	0.5936	0.6358
3	0.7566	0.6090	0.6748
4	0.7864	0.5925	0.6758
5	0.7955	0.5719	0.6654
6	0.7682	0.5719	0.6557
7	0.7709	0.6094	0.6806
8	0.7057	0.5943	0.6452
9	0.7037	0.6116	0.6544
10	0.7578	0.6436	0.6961
Avg.	0.7536	0.6042	0.6700

Table 5.4: LSTM word2vec vectors (Model-2)

LSTM model based on random vectors is used, taking 300 as the size of the random vectors. Model-1 produces the average precision, recall and F1 values of 83.76%, 30.59% and 44.71%, respectively. The complete 10-fold cross-validated results are reported in Table 5.3.

In the second round of experiments, a bi-directional LSTM model was build based on word2vec (Model-2). Here the skip-gram model was used to generate word vectors of dimension 300. As the 10-fold cross-validated results in Table 5.4 show, Model-2 achieves better performance compared to Model-1 by producing the average respective precision, recall and F-measure values of 75.78%, 60.42% and 67.00%. The precision thus actually is slightly lower than for Model-1, but the radically improved recall still makes this model perform quite a lot better overall.

In Model-3, the set of language independent features mentioned in Section 5.2.1 above (except the context feature) were used to generate feature vectors that were added to the word vectors

Fold	**Precision**	**Recall**	**F-measure**
1	0.7930	0.6667	0.7244
2	0.7016	0.6490	0.6743
3	0.8093	0.6290	0.7079
4	0.7766	0.6516	0.7087
5	0.7792	0.5974	0.6763
6	0.7614	0.6073	0.6757
7	0.7859	0.6156	0.6904
8	0.7578	0.6360	0.6916
9	0.7464	0.6467	0.6930
10	0.8055	0.6633	0.7275
Avg.	0.7717	0.6363	0.6970

Table 5.5: LSTM word2vec and features (Model-3)

built from skip-gram word2vec model. The average recall, precision and F-measure values after 10-fold cross-validation are further improved (i.e., compared both to Model-1 and Model-2):. As shown in Table 5.5, Model-3 achieves an increase both in average precision (77.2%) and recall (63.4%) compared to Model-2, for a 69.7% F1-score.

5.3 A Stacked Named Entity Recognition System for Amharic

In order to improve on the results of the previous section, a stack-based deep learning approach was used to recognize the named entities from Amharic text. Here the output of a supervised Conditional Random Field model are merged with the word vectors and fed to the LSTM model to classify the words. The pipeline of the stack-based model is shown in Figure 5.2, with the different parts further described below, starting with the supervised CRF model.

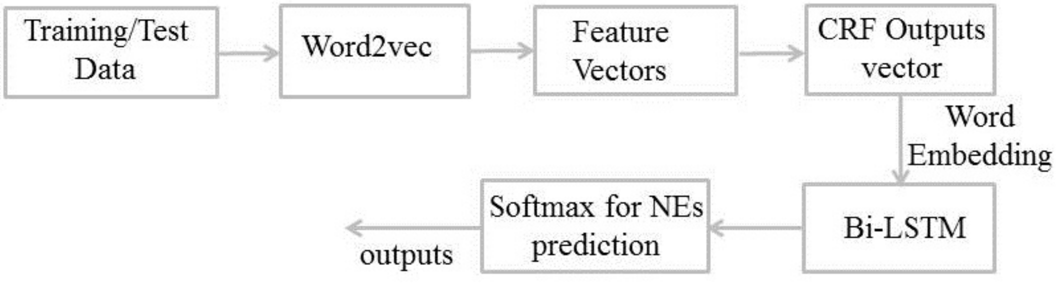

Figure 5.2: Pipeline of stack-based model

Fold	Precision	Recall	F-measure
1	0.8726	0.6887	0.7698
2	0.8070	0.6197	0.7010
3	0.8499	0.6044	0.7064
4	0.8608	0.6399	0.7341
5	0.8492	0.5580	0.6738
6	0.8335	0.6039	0.7004
7	0.8611	0.6072	0.7122
8	0.8733	0.5796	0.6968
9	0.8144	0.6133	0.6997
10	0.8804	0.6530	0.7498
Avg.	0.8502	0.6167	0.71.44

Table 5.6: CRF classifier (Model-4)

5.3.1 CRF-based model (Model-4)

In the first step, Amharic named entities are extracted using a supervised machine learning approach, namely Conditional Random Fields (CRFs). We used the C^{++} based CRF^{++} package,[4] a simple, customizable, and open source implementation of CRF for segmenting or labelling sequential data. The CRF classifier was trained on the language independent features described in Section 5.2.1.

5.3.2 Stack-based LSTM (Model-5)

The bi-directional LSTM models and word embeddings were created in the same fashion as in the previous section, but with a fifth, stack-based LSTM model being generated, where the output of the supervised model (Model-4) is used to learn a bi-directional LSTM network. Length 6 one-hot word vectors were generated (each bit represents one class) for the Model-4 outputs and concatenated with the word vectors generated by Model-2 using word2vec and the feature vectors. These three information sources are then fed into the LSTM network which classifies the tokens.

5.3.3 Results

First the supervised model (Model-4) was built using a CRF classifier based on the features mentioned in Section 5.2.1. The average 10-fold cross-validation precision, recall and F-measure values of are 85.02%, 61.67% and 71.44%, respectively. For each fold, the recall, precision and F-measure values are given in Table 5.6.

---

[4] `http://crfpp.sourceforge.net`

Fold	**Precision**	**Recall**	**F-measure**
1	0.8746	0.7267	0.7938
2	0.8048	0.6669	0.7293
3	0.8760	0.6405	0.7400
4	0.8643	0.6734	0.7567
5	0.8756	0.5847	0.7011
6	0.8431	0.6361	0.7251
7	0.8807	0.6615	0.7555
8	0.8917	0.6199	0.7313
9	0.8208	0.6722	0.7391
10	0.8658	0.6687	0.7546
Avg.	0.8597	0.6551	0.7426

Table 5.7: Stack-based LSTM (Model-5)

The stacked based model (Model-5) out-performs all previous models, with the 10-fold cross-validation results shown in Table 5.7. So reaching average precision, recall and F-measure values of 85.97%, 65.51% and 74.26%, respectively, using the outputs of the CRF learning classifier (Model-4) along with the features vectors and the word vectors from the word2vec model.

5.4 Discussion

Here we have experimented with a system for named entity recognition for Amharic, an under-resourced language. A set of language independent features was developed to extract Amharic named entities using a supervised classifier and an unsupervised bi-directional LSTM model. Better performance was achieved after creating word embeddings based on the output of the supervised model and the feature vectors together with word2vec word vectors, and then feeding the result to the neural network for training and classification.

Linguistic development is one of the most central aspects of the development of a nation. The *Ethiopian Growth and Transformation Plan* (2010–2015) also mentions language specifically: “Enhancing and promoting Ethiopian nations, nationalities, and peoples languages and folklores.” (p.75). The present work aligns with this plan, since it aims to develop means for the processing and understanding of texts written in Ethiopian languages, here focussing on Amharic which is the nation-wide communication language. Tools and resources that can help reduce language barriers and thereby provide people all over the world with improved access to information and services will have beneficial effects for most sectors of society and in the long-term contribute to the development of technology that will enable massive social and economic transformations.

The present system takes a small but important step in the direction of developing such tools. However, the error levels are still high, and many names were not identified by the system or classified into the wrong NE categories.

Many names are not identified by the system or classified into the wrong NE categories. Notably though, one error source is that many names in the training data are annotated as non-entities, but in test data the names are annotated as named entities. However, the main cause of the low recall is most likely insufficient number of training instances, which are further reduced both by Amharic being an agglutinative language and by it lacking spelling standard for many names. Poostchi et al. (2016) carried out a similar NE task on an almost equal-sized Persian corpus (250 k tokens, of which about 10% were named entities), comparing an SVM-HMM based approach to CRF and a recurrent neural network. The SVM-based classifier performed best, since their dataset also was too small for the neural network to be trained efficiently.

In the future it would be reasonable to also develop some language dependent features to improve the performance. A set of models can also be generated by using several different classifiers and ensemble these models with the help on an evolutionary algorithm. It might also be possible to utilize the word embeddings generated for Amharic in the Polyglot project. Furthermore, the word2vec model used here was built on skip-grams that predict the context words using the current word. An alternative would be to use the continuous-bags-of-words (CBOW) model, which basically does the opposite and predicts the current word from a window of surrounding context words.

Bibliography

Al-Rfou, R., V. Kulkarni, B. Perozzi, and S. Skiena (2015, June). POLYGLOT-NER: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM '15)*, Vancouver, British Columbia, Canada. Society for Industrial and Applied Mathematics.

Al-Rfou, R., B. Perozzi, and S. Skiena (2013, August). Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning (CONLL'13)*, Sofia, Bulgaria, pp. 586–594. ACL.

Alemu, B. (2013, June). A named entity recognition for Amharic. Master's thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia.

Alexander, P. (1992). Domain knowledge: Evolving themes and emerging concerns. *Educational Psychologist 27*(1), 33–51.

Androutsopoulos, J. (2011, February). Language change and digital media: a review of conceptions and evidence. In T. Kristiansen and N. Coupland (Eds.), *Standard Languages and Language Standards in a Changing Europe*, pp. 145–159. Oslo, Norway: Novus.

Auer, P. (1999). From codeswitching via language mixing to fused lects: Toward a dynamic typology of bilingual speech. *International Journal of Bilingualism 3*(4), 309–332.

Baldwin, T., P. Cook, M. Lui, A. MacKinlay, and L. Wang (2013, October). How noisy social media text, how diffrnt social media sources? In *Proceedings of the 6th International Joint Conference on Natural Language Processing*, Nagoya, Japan, pp. 356–364. AFNLP.

Belay, M. T. (2014, August). Amharic named entity recognition using a hybrid approach. Master's thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia.

Björnsson, C.-H. (1968). *Läsbarhet.* Stockholm, Sweden: Liber. (in Swedish).

Braune, F., M. Carpuat, A. Clifton, H. D. III, A. Fraser, K. Henry, A. Irvine, J. Jagarlamudi, J. Morgan, C. Quirk, M. Razmara, R. Rudinger, A. Tamchyna, and G. Foster (2012). Domain adaptation in statistical machine translation: Final report. In *CLSP Workshop Report.* Johns Hopkins University.

Bullock, B. E., L. Hinrichs, and A. J. Toribio (2014). World Englishes, code-switching, and convergence. In M. Filppula, J. Klemola, and D. Sharma (Eds.), *The Oxford Handbook of World Englishes.* Oxford, England: Oxford University Press. Forthcoming. Online publication: March 2014.

Cárdenas-Claros, M. S. and N. Isharyanti (2009). Code switching and code mixing in internet chatting: between 'yes', 'ya', and 'si' a case study. *Journal of Computer-Mediated Communication 5*(3), 67–78.

Carpuat, M. (2014). Domain adaptation in machine translation. Talk at the Machine Translation Marathon.

Carter, S. (2012, December). *Exploration and Exploitation of Multilingual Data for Statistical Machine Translation.* PhD Thesis, University of Amsterdam, Informatics Institute, Amsterdam, The Netherlands.

Chiu, J. P. C. and E. Nichols (2016). Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the ACL 4*, 357–370.

Clemen, R. T. (2008, May). Comment on Cooke's classical method. *Reliability Engineering & System Safety 93*(5), 760–765.

Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa (2011, November). Natural language processing (almost) from scratch. *J. Mach. Learn. Res. 12*, 2493–2537.

Das, A. and B. Gambäck (2013). Code-mixing in social media text: The last language identification frontier? *Traitement Automatique des Langues 54*(3), 41–64.

Das, A. and B. Gambäck (2014, December). Identifying languages at the word level in code-mixed Indian social media text. In *Proceedings of the 11th International Conference on Natural Language Processing*, Goa, India, pp. 169–178.

Debole, F. and F. Sebastiani (2005, April). An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology 58*(6), 584–596.

Deerwester, S., S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*(6), 391–407.

Demeke, G. A. and M. Getachew (2006, March). Manual annotation of Amharic news items with part-of-speech tags and its challenges. *ELRC Working Papers 2*(1), 1–17.

Eisenstein, J. (2013, June). What to do about bad language on the internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia, pp. 359–369. ACL.

Flesch, R. (1948, June). A new readability yardstick. *Journal of Applied Psychology 32*(3), 221–233.

Gafaranga, J. and M.-C. Torras (2002). Interactional otherness: Towards a redefinition of codeswitching. *International Journal of Bilingualism 6*(1), 1–22.

Gambäck, B. (2012, May). Tagging and verifying an Amharic news corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, Istanbul, Turkey, pp. 79–84. ELRA. Workshop on Language Technology for Normalisation of Less-Resourced Languages.

Gambäck, B. and A. Das (2014, December). On measuring the complexity of code-mixing. In *Proceedings of the 11th International Conference on Natural Language Processing*, Goa, India, pp. 1–7. 1st Workshop on Language Technologies for Indian Social Media.

Gambäck, B. and A. Das (2016, May). Comparing the level of code-switching in corpora. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, Portorož, Slovenia. ELRA.

Gasser, M. (2011, May). HornMorpho: a system for morphological processing of Amharic, Oromo, and Tigrinya. In *Proceedings of Conference on Human Language Technology for Development*, Alexandria, Egypt, pp. 94–99.

Geeraerts, D. (2010). *Theories of Lexical Semantics (Oxford Linguistics)*. Oxford University Press.

Genest, C. and K. J. McConway (1990, Jan/Feb). Allocating the weights in the linear opinion pool. *Journal of Forecasting 9*(1), 53–73.

Gliozzo, A. M. and C. Strapparava (2009). *Semantic Domains in Computational Linguistics*. New York, USA: Springer.

Gunning, R. (1952). *The Technique of Clear Writing*. New York, New York: McGraw-Hill.

Hu, Y., K. Talamadupula, and S. Kambhampati (2013, July). *Dude, srsly?*: The surprisingly formal nature of Twitter's language. In *Proceedings of the 7th International Conference on Weblogs and Social Media*, Boston, Massachusetts. AAAI.

Huang, Z., W. Xu, and K. Yu (2015). Bidirectional LSTM-CRF models for sequence tagging. *CoRR abs/1508.01991*.

Jamatia, A., B. Gambäck, and A. Das (2015, September). Part-of-speech tagging for code-mixed English-Hindi Twitter and Facebook chat messages. In *Proceedings of the 10th International Conference on Recent Advances in Natural Language Processing*, Hissar, Bulgaria, pp. 239–248.

Jamatia, A., B. Gambäck, and A. Das (2016, April). Collecting and annotating Indian social media code-mixed corpora. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing: Proceedings of the 17th International Conference*, Konya, Turkey. Springer.

Jiang, J. (2008). *Domain Adaptation in Natural Language Processing*. University of Illinois at Urbana-Champaign.

Kanerva, P., J. Kristofersson, and A. Holst (2000, July). Random indexing of text samples for latent semantic analysis. In *Proceedings of the 2Second Annual Meeting of the Cognitive Science Society*, Mahwah, New Jersey, pp. 1036. Lawrence Erlbaum Associates.

Karlgren, J. (1993). Sublanguages and registers - a note on terminology. *Interacting with Computers 5*, 348–350.

Kilgarriff, A. (2001). Comparing corpora. *International Journal of Corpus Linguistics 6*(1), 97–133.

Kittredge, R. (1983). *Semantic Processing of Texts in Restricted Sublanguages*, Chapter 4, pp. 45–59. International series in modern applied mathematics and computer science. Amsterdam, The Netherlands: Elsevier Science Inc.

Kudo, T. (2013). CRF++: Yet another CRF toolkit. `http://taku910.github.io`.

Lafferty, J. D., A. McCallum, and F. Pereira (2001a, June). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 1Eighth International Conference on Machine Learning*, Williamstown, Maryland, USA, pp. 282–289.

Lafferty, J. D., A. McCallum, and F. C. N. Pereira (2001b). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, San Francisco, CA, USA, pp. 282–289. Morgan Kaufmann Publishers Inc.

Lehrberger, J. (1982). Automatic translation and the concept of sublanguage. Chapter 7. Berlin, Germany: De Gruyter.

Lewis, G., B. Jones, and C. Baker (2012, October). Translanguaging: origins and development from school to street and beyond. *18*(7), 641–654.

Li, C., J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee (2012, August). TwiNER: Named entity recognition in targeted Twitter stream. In *Proceedings of the 35th International Conference on Research and Development in Information Retrieval*, Portland, Oregon, pp. 721–730. ACM SIGIR.

Lignos, C. and M. Marcus (2013, January). Toward web-scale analysis of codeswitching. In *87th Annual Meeting of the Linguistic Society of America*, Boston, Massachusetts. Poster.

Liu, X., S. Zhang, F. Wei, and M. Zhou (2011, June). Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Volume 1, Portland, Oregon, pp. 359–367. ACL.

Luckhardt, H.-D. (1991, April). Sublanguages in machine translation. In *Proceedings of the Fifth Conference of the European Chapter of the Association for Computational Linguistics*, Berlin, Germany, pp. 306–308. ACL.

Lui, M. and T. Baldwin (2014, April). Accurate language identification of twitter messages. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, Göteborg, Sweden, pp. 17–25. ACL. 5th Workshop on Language Analysis for Social Media.

Lyu, D.-C., T.-P. Tan, E.-S. Chng, and H. Li (2015, September). Mandarin–English codeswitching speech corpus in South-East Asia: SEAME. *Language Resources and Evaluation 49*(3), 581–600.

Magnini, B. and G. Cavaglià (2000, May). Integrating subject field codes into wordnet. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, Athens, Greece. ELRA.

Mehamed, M. A. (2010, November). Named entity recognition for Amharic language. Master's thesis, Department of Computer Science, Addis Ababa University, Addis Ababa, Ethiopia.

Melby, A. (1997). Some notes on the proper place of men and machines in language translation. *Machine Translation 12*(1/2), 29–34.

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., I. Sutskever, K. Chen, G. Corrado, and J. Dean (2013). Distributed representations of words and phrases and their compositionality. *CoRR abs/1310.4546*.

Moen, H., E. Marsi, and B. Gambäck (2013, August). Towards dynamic word sense discrimination with random indexing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, pp. 83–90. ACL. Workshop on Continuous Vector Space Models and their Compositionality.

Molina, G., F. AlGhamdi, M. Ghoneim, A. Hawwari, N. Rey-Villamizar, M. Diab, and T. Solorio (2016, November). Overview for the second shared task on language identification in code-switched data. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas. ACL. 2nd Workshop on Computational Approaches to Linguistic Code Switching.

Moro, A., A. Raganato, and R. Navigli (2014). Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computional Linguistics 2*, 231–244.

Muysken, P. (2000). *Bilingual speech: A typology of code-mixing.* Cambridge, England: Cambridge University Press.

Navigli, R. and S. P. Ponzetto (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence 193*, 217–250.

Nguyen, D. and L. Cornips (2016, August). Automatic detection of intra-word code-switching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 82–86. ACL. 14th Annual SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology.

Nguyen, D. and A. S. Doğruöz (2013, October). Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, pp. 857–862. ACL.

Paolillo, J. (1996, June). Language choice on soc.culture.punjab. *Electronic Journal of Communication 6*(3).

Paolillo, J. (1999, June). The virtual speech community: Social network and language variation on IRC. *Journal of Computer-Mediated Communication 4*(4).

Paolillo, J. (2011, June). "conversational" codeswitching on usenet and internet relay chat. *Language@Internet 8*(article 3).

Partee, B. H. (1976, January). *Montague Grammar.* New York, New York: Academic Press.

Pinto, D., P. Rosso, and H. Jiménez-Salazar (2011, July). A self-enriching methodology for clustering narrow domain short texts. *The Computer Journal 54*(7), 1148–1165.

Plank, B. (2011). *Domain Adaptation for Parsing.* Ph.d. thesis, University of Groningen.

Poostchi, H., E. Z. Borzeshi, M. Abdous, and M. Piccardi (2016, December). PersoNER: Persian named-entity recognition. In *Proceedings of the 26th International Conference on Computational Linguistics*, Osaka, Japan, pp. 3381–3389. ACL.

Ramage, D., D. Hall, R. Nallapati, and C. D. Manning (2009, August). Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, pp. 248–256. ACL.

Ritter, A., S. Clark, Mausam, and O. Etzioni (2011, August). Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, pp. 1524–1534. ACL.

Rizzo, G., M. van Erp, J. Plu, and R. Troncy (2016, April). Making sense of microposts (#Microposts2016) Named Entity rEcognition and Linking (NEEL) challenge. In *Proceedings of the 6th Workshop on Making Sense of Microposts (#Microposts2016)*, Montréal, Canada, pp. 50–59.

Rong, X. (2014). word2vec parameter learning explained. *CoRR abs/1411.2738*.

Sikdar, U. K. and B. Gambäck (2016a, December). Feature-rich Twitter named entity extraction and classification. In *The 2nd Workshop on Noisy User-generated Text (W-NUT) at the 26th International Conference on Computational Linguistics (COLING'16)*, Osaka, Japan. ACL.

Sikdar, U. K. and B. Gambäck (2016b, November). Language identification in code-switched text using Conditional Random Fields and Babelnet. In *Proceedings of the 2nd Workshop on Computational Approaches to Code Switching at the 2016 Conference on Empirical Methods on Natural Language Processing (EMNLP)*, Austin, Texas. ACL.

Sikdar, U. K. and B. Gambäck (2016c, December). Twitter named entity extraction and linking using differential evolution. In *Proceedings of the 13th International Conference on Natural Language Processing (ICON'16)*, Varanasi, Uttar Pradesh, India, pp. 198–207.

Solorio, T., E. Blair, S. Maharjan, S. Bethard, M. Diab, M. Gohneim, A. Hawwari, F. Al-Ghamdi, J. Hirschberg, A. Chang, and P. Fung (2014, October). Overview for the first shared task on language identification in code-switched data. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp. 62–72. ACL. 1st Workshop on Computational Approaches to Code Switching.

Solorio, T. and Y. Liu (2008, October). Learning to predict code-switching points. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Honolulu, Hawaii, pp. 973–981. ACL.

Storn, R. and K. Price (1997, December). Differential Evolution — a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization 11*(4), 341–359.

Strauss, B., B. E. Toma, A. Ritter, M. C. de Marneffe, and W. Xu (2016, December). Results of the WNUT16 named entity recognition shared task: Twitter lexical normalization and named entity recognition. In *Proceedings of the 26th International Conference on Computational Linguistics*, Osaka, Japan, pp. 138–144. ACL. 2nd Workshop on Noisy User-generated Text.

Takeuchi, K. and N. Collier (2002). Use of support vector machines in extended named entity recognition. In *Proceedings of the 6th Conference on Natural Language Learning - Volume 20*, COLING-02, Stroudsburg, PA, USA, pp. 1–7. Association for Computational Linguistics.

Trier, J. (1973). Das sprachliche felt: eine auseinandersetzung. In *Neue Jahrbücher für Wissenschaft und Jugendbildung*, Volume 10, pp. 428–49. The Hauge, The Netherlands: Mouton & Co.

Voss, C., S. Tratz, J. Laoudi, and D. Briesch (2014, May). Finding romanized Arabic dialect in code-mixed tweets. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, Reykjavík, Iceland, pp. 188–199. ELRA.

Waitelonis, J. and H. Sack (2016, April). Named entity linking in #tweets with KEA. In *Proceedings of the 6th Workshop on Making Sense of Microposts (#Microposts2016)*, Montréal, Canada, pp. 61–63.

Zhou, G. and J. Su (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL ’02, Stroudsburg, PA, USA, pp. 473–480. Association for Computational Linguistics.