LOW-RESOURCE NAMED ENTITY RECOGNITION

Stephen Mayhew

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2019

Dedicated to Heidi.

ACKNOWLEDGEMENT

This PhD has never been a solo effort, and my life has been touched and enriched by many people. There are far too many to name, but I can at least mention a few.

My advisor, Dan Roth. One of the hallmarks of working with Dan is that he always treats you as a colleague. I remember leaving meetings feeling as though I had discussed ideas with an equal, not as one who has Experience and Citations and Tenure at an Important University. In several group meetings, I noticed that if someone made a point that was clearly wrong, Dan would never shoot it down, but rather find a kernel of interest in it.

Transferring between schools mid-degree is sometimes considered a worst-case scenario for a grad student, but Dan fought for us to ensure that it was a smooth transition. Classes were transferred, offices were negotiated, and he even offered to help with insurance when it became clear that the cost was significantly more than in Illinois. Set down this: I would do it again. I often think that it says something about him as an advisor that nearly all of his students chose to move across the country to stay with him.

My many colleagues and collaborators over the years: Chen-Tse Tsai, Shyam Upadhyay, Daniel Khashabi, Nitish Gupta, Dan Deutsch, Tatiana Tsygankova, Snigdha Chaturvedi, Mark Sammons, Subhro Roy, Robert Shaffer, Jordan Kodner, Reno Kriz, Jennifer Sheffield, Xiaodong Yu, Chase Duncan, Sihao Chen, Xuanyu (Ben) Zhou, Christos Christodoulopoulos, Parisa Kordjamshidi, Anne Cocos, Daphne Ippolito, João Sedoc, Colin Graber, Yangqiu Song, Eric Horn, Alice Lai, Jason Rock, Daphne Tsatsoulis, Yonatan Bisk, Chris Cervantes, Ryan Musa, Rob Deloatch. Thanks for the many conversations over lunch or coffee, foosball games, game nights, walks, conference trips, late night submissions, rejection commiserations and acceptance celebrations, and many other things.

Those who welcomed me into the Cognitive Computation Group: Vivek Srikumar, V.G. Vinod Vydiswaran, Kai-Wei Chang, not to mention the many others I've met since.

The numerous interns working on the LORELEI project in 2018 and 2019. Those I worked most closely with were Tatiana Tsygankova, Yihao Chen, Lucia Huo, Devanshu Jain, Francesca Marini, Zihan Wang, and Jane Lee. Our success in these evaluations couldn't have happened without them, and I'm grateful to all of them for their eager attitudes, openness to learning new things, and hard work.

Dean and Deb at Magic Carpet. I don't know your last name, but I always appreciated the side of good humor served with lunch.

Professors at Rose-Hulman Institute of Technology, my undergraduate alma mater, who originally encouraged me to go into grad school. Matt Boutell, who taught a computer vision class and told me he would have taken me as a student, and Claude Anderson, who told me I would make a good grad student. They also gave me advice to apply as a PhD student even if I thought a Master's might be the terminal point. It worked out alright, but I should have taken their advice.

Many senior researchers who have given good advice and shown kindness: Julia Hockenmaier, Ben Van Durme, Hoifung Poon, Jerry DeJong, Mark Dredze, Chris Callison-Burch, Mitch Marcus, Chris Quirk, Wen-tau (Scott) Yih, Kristina Toutanova, Ming-wei Chang.

The many staff who helped things run smoothly. In particular, Kathy Runck, Dawn Cheek, Eric Horn, and the others in the Siebel Center at the University of Illinois. Bus days with Miss Kathy were a highlight for myself and my kids. Those at University of Pennsylvania included Mike Felker, Cheryl Hickey, Jennifer Sheffield, and Lily Hoot.

David Suryk and many good friends we made at the Graduate Christian Fellowship at the University of Illinois.

The third spaces in our lives: our church families at Hessel Park Christian Reformed Church in Champaign, IL, and City Church, in Philadelphia, PA.

Finally, my family: Heidi, Alan, Lois, and Phoebe. I never felt lonely with you around.

ABSTRACT

LOW-RESOURCE NAMED ENTITY RECOGNITION

Stephen Mayhew

Dan Roth

Most of the success in natural language processing (NLP) in the last 20 years has come from statistical machine learning methods that discover complex patterns in text and make predictions. These methods traditionally require supervised data, which is nearly always created by humans, as a gold standard for that task. But as we look to extend these successes to other languages, we are faced with the daunting task of starting from scratch. The years of effort that went into creating annotations for English and a select few popular languages must be relived for each new language. This unrealistic requirement means that as we seek to perform old tasks in new languages we must use existing resources, or rapidly develop new resources.

In particular, we study the problem of Named Entity Recognition (NER) in low resource languages. The task of NER is to find and classify names in text, and the low-resource qualifier signifies that we build these models without access to training data. This thesis discusses the use of incidental signals for developing NER systems, such as character sequences indicative of named entities, or partially-annotated text, such as might come from non-speaker annotations. It describes new methods for cross-lingual NER, exploiting such resources as Wikipedia and bilingual lexicons. The penultimate chapter applies several prominent techniques to a broad array of test languages, giving valuable insights into what has been accomplished, and what is left to do. The final chapter distils knowledge from several years of experience building low-resource NER systems into a practical guide.

List of Tables

List of Figures

PUBLICATION NOTES

1. Chen-Tse Tsai, Stephen Mayhew, Dan Roth. Cross-lingual Named Entity Recognition via Wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL), 2016.* URL `https://www.aclweb.org/anthology/K16-1022`

2. Stephen Mayhew, Chen-Tse Tsai, Dan Roth. Cheap Translation for Cross-lingual Named Entity Recognition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2017.* URL `https://www.aclweb.org/anthology/D17-1269`

3. Stephen Mayhew, Snigdha Chaturvedhi, Chen-Tse Tsai, Dan Roth. Named Entity Recognition with Partially Annotated Training Data. *Proceedings of The 23th SIGNLL Conference on Computational Natural Language Learning (CoNLL), 2019.*

4. Xiaodong Yu, Stephen Mayhew, Mark Sammons, Dan Roth. On the Strength of Character Language Models for Multilingual Named Entity Recognition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018.* URL `https://www.aclweb.org/anthology/D18-1345`

5. Stephen Mayhew, Dan Roth. TALEN: Tool for Annotation of Low-resource ENtities. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics-System Demonstrations (ACL) 2018.* URL `https://www.aclweb.org/anthology/P18-4014`

CHAPTER 1 : Introduction

Most work in natural language processing (NLP) is done in English, the *lingua franca* of the scientific world. As a consequence, a massive ecosystem of English resources has developed. Primarily, these resources are in the form of supervised annotations, such as Penn Treebank (Marcus et al., 1993) or OntoNotes (Pradhan et al., 2007), but also include tremendous amounts of raw text, and an abundance of English-speaking annotators.

To a lesser extent, similar resource ecosystems have developed around other popular languages, such as German, Spanish, and Chinese, but for most of the world's roughly 7000 languages,[1] there exist nearly no resources. In the data-driven world of machine learning, and especially with the popularity of data-hungry neural methods, this lack of resources means that most languages have no suitable NLP tools.

The field of low-resource NLP addresses this shortcoming by targeting languages with little to no resources. Common techniques in this field are to exploit resources from some high-resource language, or to rapidly develop resources directly in the target language. My particular focus in this thesis is on low-resource methods for named entity recognition (NER), the task of discovering and classifying named entities (for example, persons, organizations, locations) in text, as in Figure 1. While there has been plenty of work on NER in situations with gold annotated training data, I study methods for dealing with situations with minimal or non-existent training data.

Noting that the top 94 languages cover 80% of the world's speakers,[2] one may wonder if research efforts in this direction are well-spent. To answer this, we look to a 2003 recommendation from the United Nations Educational, Scientific, and Cultural Organization (UNESCO) concerning "the Promotion and Use of Multilingualism and Universal Access to Cyberspace" (UNESCO, 2003). In this recommendation, UNESCO addresses many issues

[1] `https://www.ethnologue.com/guides/how-many-languages`

[2] From Ethnologue 22nd edition, accessed Aug 2, 2019 `https://www.ethnologue.com/statistics/size`

Amsterdam GPE officials say that Dick Van Dyke PER will no longer be allowed in the country.

It's a grade-A gray day Gray Davis PER .

In July 2006, Arsenal F.C. ORG moved into the Emirates Stadium LOC , after 93 years at Highbury LOC .

Figure 1: Examples of sentences annotated with named entities. The tags represented in this set are Geo-Political Entity (GPE), Person (PER), Location (LOC), and Organization (ORG).

surrounding language use on the internet, including national language policy, education and literacy policy, access to network infrastructure, and, most relevant here, development of content and systems in all languages, including indigenous and minority languages. Progress in these areas, they argue, directly advances one of the key purposes of UNESCO, as stated in their constitution: “to promote the free flow of ideas by word and image.”[3] Of course, for many languages *system and content development* is held back by more basic problems, such as the lack of tools like keyboards and appropriate fonts. But these hurdles will eventually be passed, and ultimately access to structured information from unstructured text will require powerful NLP tools, as we have seen in English.

But there’s another crucial use case, which is the need for those who speak one language to understand text in other languages, which we may call cross-lingual language understanding. The internet largely consists of unstructured text, containing an immense amount of information waiting to be extracted. In English, the NLP community has made great strides with decades of work on information extraction (Riloff et al., 1993; Grishman, 1997) and other tasks, in which meaningful structured information is gathered from unstructured text. But most of the internet is not in English,[4] which means that most of the unstructured information on the internet is not accessible with today’s NLP tools.

For a humanitarian example, consider the situation faced by researchers at Microsoft during

[3]Article I, paragraph 2(a)

[4]`https://en.wikipedia.org/wiki/Languages_used_on_the_Internet`

the earthquake in Haiti in 2010 (Lewis, 2010). Aid organizations had set up hotlines to receive text messages with requests for help, but the thousands of texts that flooded in were sent in Haitian Creole, a low-resource language. In order to properly triage these messages, it was important to translate into English.

In addition to humanitarian goals, there is a great commercial interest in multilingual systems. Clientele may span multiple countries and languages, so the ability to process and understand more languages translates directly to more customers.

For the above reasons, I argue that multilingual NLP systems are a key ingredient in the global information economy. With this understanding, the key question is how to go about developing such systems? An engaged reader may ask two sensible questions: 1) why not translate all text into English and apply English tools? and 2) why not invest time and money into building language-specific resource ecosystems?

Why not translate? In this hypothetical scenario, the goal is to make some NLP predictions on text in a foreign language. But instead of using tools in that language, we translate into English, make predictions on the text, and translate back again, perhaps associating the annotations with the foreign text using translation alignments.

An immediate problem with this approach is that machine translation systems require millions of lines of parallel text for training. In the Fourth Conference on Machine Translation (WMT19), the German-English training data for the News Translation shared task consisted of nearly 40 million parallel sentences (Barrault et al., 2019).

But even with this training data, and despite recent breakthroughs (Wu et al., 2016; Bojar et al., 2016), machine translation remains an extremely hard problem (Koehn and Knowles, 2017). Low-resource machine translation (Zoph et al., 2016), in which the available parallel text is measured in the thousands of lines, is even more difficult. We cannot expect that automatic translation, high- or low-resource, will preserve all the elements needed for successful annotation in English.

In a sense, machine translation is the hardest NLP task. This is to say that a perfect machine translation system needs to accomplish a large array of other NLP tasks (perhaps implicitly) in order to translate well. For example, since we study NER, consider the following (adversarial) sentence: “his favorite artist was queen”, which contains the entity “queen”, referring to the rock band fronted by Freddie Mercury. Google Translate[5], targeting German (a high-resource language), translates this as “sein Lieblingskünstler war Königin.” This is mostly correct, except that the entity “queen” is translated into “Königin”, the German word for “queen.” But that’s not how named entities tend to work – the band Queen is Queen wherever they go. In order to correctly translate this sentence, Google Translate needs to do NER.[6] It makes no sense to require a harder task (MT) to solve an easier task (NER).

Why not build language-specific ecosystems? If we cannot rely on machine translation, perhaps the best course of action is to buckle down and develop resources in every language. After all, as a rule of thumb, monolingual systems are almost always better than cross-lingual systems.

This is an admirable project, and greatly to be commended. Increasing the diversity of language resources is scientifically valuable in that it allows a broader exploration of what NLP tools can learn. After all, we can hardly claim to have achieved Natural Language Understanding if it is just English Language Understanding. Such resources are also practically valuable for the reasons outlined above.

But the difficulty of this, of course, is the scale. It has taken many years to build up strong NLP resources and models in English. Repeating this effort even for the most popular 100 languages could take decades (depending on the level of annotation desired). At the same time, task definitions change and domains shift, so new datasets become old datasets, and

[5]`translate.google.com`, accessed Aug 2, 2019

[6]If the sentence was capitalized correctly, Google Translate knows what to do, but this is an artifact of NER being easy in English.

the problem is never solved. In the meantime, we need to have tools for low-resource NLP.

We approach the problem of low-resource NER with two different paradigms. In the first, we explore the use of incidental signals for NER. In the second paradigm, we exploit annotated data from high-resource languages using cross-lingual methods.

We use the term 'incidental' after Roth (2017), to signify features or patterns in the data that fall short of supervised annotations, but can nonetheless be used in learning a target distribution. First, we give some details on a tool for low-resource NER annotation (Talen), and follow with experiments that quantify the quality of non-speaker annotators as compared to native speakers. Next, we study how character sequences in named entities are distinct from those in non-named entities. Intuitively, this explains how, at least in English, some words *sound* like names, and others sound like regular words. We showed how this intuition can be exploited in a simple fashion to improve NER results. Another signal we studied was that of partial NER annotations, as might be produced from a set of rules, or an annotator without knowledge of the target language.

In one work (Section 4.1), we developed language-invariant features based on Cross-lingual Wikification, which in turn is based on inter-language links in Wikipedia. This allowed us to train a model using English training data and transfer it directly to target languages, relying on the Wikipedia-based language-invariant features. One limitation of this work is that it depends on the size of the target language Wikipedia, and the number of inter-language links. As it happens, at the time of writing, there are relatively few languages with Wikipedias large enough to be useful. For example, the Wikipedia for Uyghur has 4,137 articles,[7] and a relatively small intersection of inter-language links. This fact led us to develop a model requiring leaner resources. Observing that NER models focus strongly on surface forms, and features for NER systems are often somewhat shallow, we proposed a 'cheap translation' method. In this method, annotated text in a source language (usually English) is "translated" into the target language word for word. This annotated "target"

[7]`https://en.wikipedia.org/wiki/List_of_Wikipedias`, accessed Aug 2, 2019

language text can then be used as supervision in a standard model. Obviously, this produced poor quality translations, but gave enough traction for models to learn something valuable.

In the penultimate chapter, we evaluate several of the more popular algorithms on a multilingual dataset of 23 languages. This analysis shows interesting differences between languages in the study, as well as improvements in recent models. This study provides a baseline for the many languages, and points a way forward for future work in cross-lingual methods.

The final chapter is a record of some of the expertise we have gathered from four years of surprise language evaluations. We have documented several of the practices that worked well for us, including a practical protocol for building a low-resource NER system. We also include our results from each evaluation as a guide for what to expect.

While we focus on NER in this thesis, we treat it as representative of a much larger ecosystem of NLP tools and research. First, because NER is a vital upstream step in many NLP pipelines, including relation extraction, semantic parsing, and question answering. But also in the sense that the task of NER is a relatively simple proxy or testbed for cross-lingual techniques. We hope that the experiments in cross-lingual learning in this field will transfer to other fields as well.

1.1. Thesis Statement

In this thesis, I argue that NER systems can be built even in the absence of target language training data. By taking advantage of cheap and easily available resources, such as Wikipedia, lexicons, and English speakers, and by using inexpensive and incidental signals, we can either transfer knowledge from high-resource languages, or develop annotations directly in low-resource languages.

1.2. Outline of This Work

Following a chapter covering background and related work, this thesis will contain chapters with the following content:

Indirect signals for low-resource NER Although explicit supervised annotations may be unavailable in certain situations, it is often possible to use incidental signals to build NER systems. We introduce a tool called TALEN that helps a non-speaker of a language annotate text with NER tags, and quantify this process with human experiments. We study such signals as word-internal character sequences, and show a powerful method for learning from partially-annotated sequences.

Cross-lingual NER Most situations of low-resource NER come from languages in which there is no annotated data available (and there are thousands of such languages). This chapter explores some cross-lingual methods for building systems in low-resource languages, first by using the rich multilingual structure of Wikipedia, and then by using readily available lexicons to "translate" training data into a target language.

Massively Multilingual Analysis of NER In the course of cross-lingual research, it is not uncommon to choose languages which look good for your method. In this section, we survey and evaluate several of the most common cross-lingual methods available as of writing, and make observations on relative quality, as well as suggestions for future work.

How to Build a Surprise-language NER System in One Week The thesis will conclude with a chapter that ties up loose ends, and looks toward future work.

CHAPTER 2 : Background and Related Work

This chapter will outline the task of named entity recognition, and move on to discuss related work in low-resource topics.

2.1. Named Entity Recognition

Named Entity Recognition (NER), sometimes also called named entity recognition and classification (NERC), is the task of discovering and classifying named entities in text. The definition of "named entity" varies across datasets, but some common types are person (PER), organization (ORG), and location (LOC). Below are some examples of phrases annotated with named entities (the last example is in German).

- My sister [Paris PER] lives in [Lawrence LOC].
- Marathoner [Eliud Kipchoge PER] honored by the [University of Pennsylvania ORG].
- Es gibt viele Menschen in [Berlin LOC].

When building an NER system, one typically trains a **machine learning model** on some large **corpus of text annotated with named entities**. To understand the task as a whole, we must understand these two components. But before addressing these, we ask two important questions: why is NER hard? And why is NER important?

Why is NER hard? Although certain names are common and therefore frequently attested in corpora, one difficulty of this task is that names form an open class of words. One can never assume that a rule list or training corpus contains all possible names. After all, unless you, dear reader, follow long-distance running, you have probably never heard of Eliud Kipchoge in the example above, or even seen these individual tokens in other contexts. Locations and organizations are also open class: eSwatini is an example of a recently coined location, and Netflix and Duolingo are recently coined organization names.

So in order to detect named entities reliably, systems must rely on context of the text. In English, and many other Latin- or Cyrillic-script languages, entities are conventionally marked with capitalization, which makes detection substantially easier. But to simulate the general situation in which no capitalization is available, consider the following lowercased English examples:

- in the marathon event, [hall PER] outran [huddle PER] to win first place.
- his name was [hoekstenberger PER], and he came from [etwaburg LOC].

In the first example, there are two people mentioned (real-life marathoners Sara Hall and Molly Huddle), and each has a last name that is also a common English word. Despite the adversarial nature of the example, most human annotators would still correctly recognize the names. What sort of reasoning is required? Perhaps a human recognizes these common words as names because both “hall” and “huddle” lack determiners. Further, perhaps there is the understanding that the arguments to the verb “outran” tend to be animate objects, and that only humans run marathons.

In the second example, “hoekstenberger” and “etwaburg” are neologisms (I coined them), but you can identify them as names in two ways: they just look like names (as we explore later Section 3.3), and the context “his name was” and “came from” are dead giveaways.

Why is NER important? From a practical point of view, there are many NLP tasks which use output from NER. For example:

- In machine translation (Nikoulina et al., 2012; Ugawa et al., 2018), names can be ambiguous (for example, Rose Keys), and should also be kept contiguous in translation.
- In question answering and knowledge base completion, answers to questions are often named entities (Mollá et al., 2006; Yao et al., 2013; Das et al., 2017; Sun et al., 2019).
- In relation extraction (Zelenko et al., 2003; Lin et al., 2016), arguments to relations (spouse of, president of, born in) are nearly always named entities.
- In entity linking (Gupta et al., 2017), detection of named entities is an explicit preprocessing step.

- In summarization, names that are mentioned are often related to the topic at hand, if not integral to the summary. For example, a summarization of a tennis match should include which players were playing, where they played, and perhaps even names of famous guests (Wolfe et al., 2018; Amplayo et al., 2018).
- When doing spell check, it can be frustrating or even insulting to people with non-Western names if their names are never accepted, or if alternate suggestions are given. A spell check system that detects names and declines to offer suggestions could solve this problem (Ruch et al., 2003; Ramerth et al., 2012).

In a more philosophical sense, language understanding requires knowledge of named entities. Deictic expressions ("that place", "this child") may be useful in personal communication, but in text that assumes no shared context (arguably the primary domain of NLP), the use of named entities as reference is unavoidable. Imagine for a moment a news article that contained no named entities. Certain types of articles, such as opinion pieces, recipes, poetry, or other abstract forms may succeed without named entities, but most everything else needs to be grounded in order to convey information. For example, consider the following paragraph, which has all named entities removed:

> The ORG on Friday approved the merger of ORG and ORG, the third- and fourth-largest wireless companies in the LOC, moving the deal one crucial step closer to completion years after the two carriers first explored joining forces.[1]

Without the named entities, one might understand that the article discusses a business deal relating to wireless companies, but without prior knowledge of this event, it's not even clear which country this is happening in. It could be any country that has 4 or more wireless companies.

Imagine then for a moment the opposite: the same news article that has all but the named entities removed. These entities are: Justice Department, T-Mobile, Sprint, United States.

[1]Taken from the New York Times, July 26th, 2019

Naturally, neither example gives the entire picture, but the list of names is more specific. We can't understand text without understanding the named entities present in it.

2.1.1. Models

As with nearly all NLP tasks, the space of NER models can be divided into non-neural and neural architectures. We defer a comprehensive literature review to other publications (Sekine, 2004; Nadeau and Sekine, 2007; Yadav and Bethard, 2018; Li et al., 2018), but give a brief overview of popular architectures.

Non-neural Methods

NER systems built before the neural age were essentially word classification models, with features drawn from a fixed size context window around each target word. The exact features used would vary by implementation, but the most common included, for each word in a context window (and often concatenated with the position relative to the target word):

- Surface form of word
- Word shape (such as Aaaa, AAA, aaaa)
- Whether or not the word is capitalized
- Character n-grams
- Part of Speech
- Brown cluster bit string (possibly a prefix of the bit string)
- Gazetteer features, indicating whether or not a word appears in a pre-defined list of entities

Many different kinds of models were used, including Support Vector Machines (SVM) (McNamee and Mayfield, 2002), decision trees with AdaBoost (Carreras et al., 2002), averaged perceptron (Ratinov and Roth, 2009), Hidden Markov Models (HMM) (Burger et al., 2002), and Conditional Random Fields (CRF) (Lafferty et al., 2001; McCallum and Li, 2003; Finkel et al., 2005).

One important paper in the field was Ratinov and Roth (2009), which identified 4 key design decisions in the development of an NER system, and experimented with variations on each decision. These 4 design decisions were related to chunk representation (BILOU vs BIO), inference[2] (greedy vs Viterbi), use of non-local features, and use of external resources. This led to strong performance on CoNLL2003 English data, which would remain state-of-the-art performance for many years.

There were two important pieces of software for non-neural NER: CogCompNLP[3] (Ratinov and Roth, 2009; Khashabi et al., 2018a), and Stanford NER[4] (Manning et al., 2014). Both systems were written in Java, and were relatively easy to download and use, leading to widespread adoption. CogCompNLP is an implementation of Ratinov and Roth (2009), and Stanford NER is most closely related to Finkel et al. (2005).

Given the popularity of CRFs in NER, we give a brief overview, specifically linear-chain CRFs, borrowing notation from Sutton et al. (2012), Definition 2.1. Given a parameter vector $\theta = \{\theta_k\} \in \mathcal{R}^k$, we can calculate the probability of a label sequence $\mathbf{y} = \{y_1, y_2, ..., y_T\}$ conditioned on a word sequence (sentence) $\mathbf{x} = \{x_1, x_2, ..., x_T\}$.

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(X)} \prod_{t=1}^{T} \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\} \tag{2.1}$$

Where $\{f_k(y_t, y_{t-1}, \mathbf{x}_t)\}_{k=1}^{K}$ are real-valued feature functions, and $Z(\mathbf{x})$ is a normalization function, defined as:

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{t=1}^{T} \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\} \tag{2.2}$$

Where $\mathbf{y}$ underneath the sum represents *all possible sequences* of labels.

[2] Or *decoding*

[3] `https://github.com/CogComp/cogcomp-nlp`

[4] `https://nlp.stanford.edu/software/CRF-NER.shtml`

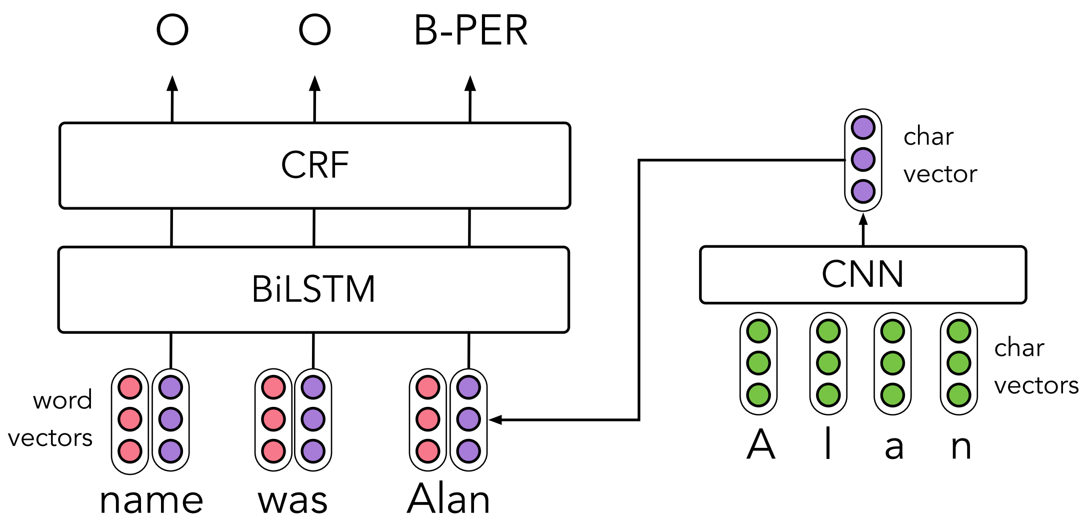


Figure 2: Diagram of the standard BiLSTM-CRF model (Ma and Hovy, 2016), with CNN for character embeddings.

To describe this in natural language, this equation represents some score of a sequence of labels paired with a sentence of T tokens. During optimization, we update the parameters of the model to fit known sequences of labels. At test time, we can infer the optimal sequence of labels. The normalization may seem intractable because of the way that it marginalizes over all possible sequences, but in practice this can be computed efficiently with dynamic programming.

In practice, these feature functions $f_k(y_t, y_{t-1}, \mathbf{x}_t)$ are defined as described above. In principle, feature functions in a CRF are very expressive in that they can be conditioned on current and prior tag assignments in addition to tokens.

Neural Methods

The most common deep learning architecture for NER is the bidirectional long short-term memory network (LSTM) (Hochreiter and Schmidhuber, 1997) with conditional random field (BiLSTM-CRF) proposed in Ma and Hovy (2016), but popularized by Lample et al.

(2016), in part because of an accessible implementation.[5] In this model, a sentence is encoded with a bidirectional Long Short Term Memory (BiLSTM) network, preserving all hidden states. These hidden states are then used as input features to a CRF, which calculates the most probable output sequence. Given the importance of surface forms in NER, it is also common to encode the characters of each word (often with a Convolutional Neural Network (CNN)) which representation is then concatenated with the corresponding word embeddings. These character-level embeddings are crucial in situations where there are many OOV tokens, or when the word embeddings are lower-cased.

Although the BiLSTM-CRF is most popular, other architectures exist. Strubell et al. (2017a) propose a model based on iterated dilated convolutions, allowing for far more efficient training while sacrificing no performance. In addition to the BiLSTM-CRF architecture, Lample et al. (2016) proposed a transition-based chunking model reminiscent of models for dependency parsing. Devlin et al. (2019) fine-tune a deep pretrained transformer with an added linear layer for strong performance in NER. For a more comprehensive survey of deep learning techniques for NER, see (Yadav and Bethard, 2018).

2.1.2. Data

On the data side of NER, we discuss common corpora, their domains and tagsets, and finish with evaluation metrics.

Corpora

In English, arguably the most popular dataset is CoNLL 2003 English (Tjong Kim Sang and De Meulder, 2003), which was created for a shared task on language independent NER for the Conference on Natural Language Learning (CoNLL) in 2003. This dataset consists of documents taken from the Reuters RCV1 corpus (Rose et al., 2002), and annotated with no overlap by the two organizers of the shared task, neither of whom are native speakers of

[5] `https://github.com/glample/tagger`

English.[6] The tagset consists of 4 coarse-grained tags: person, organization, location, and miscellaneous. The first 3 are self-explanatory, but the final tag is a broadly-defined catch-all that contains such entity classes as event, language, nationality, and product. While the domain is ostensibly newswire, there are also many documents that report tabular statistics, such as sports scores, or trade reports. Likely because of these entity-dense documents, CoNLL 2003 English has an unusually high ratio of entity tokens to non-entity tokens (Augenstein et al., 2017).

CoNLL 2003 English has been the primary benchmark for NER ever since its creation. At the original evaluation, scores on the test set (testb) were around 88 F1 (Florian et al., 2003), and as of writing, state of the art scores are around 93.5 (Clark et al., 2018). These high scores have led some to say that NER is "solved", although this ignores idiosyncracies of the dataset, such as a high number of shared entities between the train and test set (Augenstein et al., 2017), and the prevalence of proper capitalization (Mayhew et al., 2019c).

In addition to English, there is a German corpus released in CoNLL 2003, and Dutch and Spanish corpora released for CoNLL 2002 (Sang, 2002). All 4 datasets use the same tagset, and are often collectively referred to as the CoNLL NER dataset. While it is not uncommon for researchers to evaluate models on all 4 datasets, the English dataset is vastly more popular than the others.

Another popular NER dataset is OntoNotes (Hovy et al., 2006; Pradhan et al., 2007), with a rich set of annotations for English, Arabic, and Chinese. As with CoNLL, the English dataset is most widely used. OntoNotes differs from CoNLL in several important ways. First, the tagset contains 18 types, including the common types of person, and organization, and location (albeit defined differently), and new types such as numbers and dates. The annotation guidelines are also different in several important ways, making transfer between the datasets tricky. OntoNotes is also an order of magnitude larger, and is divided according to several diverse genres, including newswire, transcribed telephone conversa-

[6]Personal communication with Erik Tjong Kim Sang.

tions, and magazines. Different test splits of OntoNotes exist, most notably the version 4 split, as used in Ratinov and Roth (2009), and the version 5 split defined in Pradhan et al. (2011) and as used in Durrett and Klein (2014) and Strubell et al. (2017b). OntoNotes is often used in addition to CoNLL 2003 English in order to better show generality.

There also exist datasets in specialized domains, such as twitter (Derczynski et al., 2016, 2017), code-switched data (Aguilar et al., 2018), or biomedical applications (Kim et al., 2003; Tanabe et al., 2005). Other variations on the NER task also exist, such as fine-grained NER (Ling and Weld, 2012a) and nested NER (Finkel and Manning, 2009). In this thesis, we target only non-nested coarse-grained NER.

There are many one-off projects to build NER datasets in other languages, for example in Arabic (Mohit et al., 2012), Bengali (Ekbal and Bandyopadhyay, 2008), Hungarian (Szarvas et al., 2006), Romanian (Mitrofan, 2017), Chinese (Shih et al., 2004), Persian (Farsi) (Poostchi et al., 2018), Japanese (Iwakura et al., 2016), or Czech (Kravalová and Žabokrtský, 2009). However, these may suffer from varying annotation quality, size, and availability.

Similar to the CoNLL 2002 and 2003 workshops, there have been shared tasks for NER in other languages. As part of the Workshop on NER for South and South East Asian Languages at IJCNLP 2008, there was an NER shared task with corpora for Hindi, Bengali, Oriya, Telugu and Urdu (Singh, 2008). The Balto-Slavic NLP workshop at ACL 2019 had a shared task on NER for Bulgarian, Czech, Polish, Russian (Piskorski et al., 2019).

The largest coordinated project for multilingual NER corpora is from the LORELEI project (Christianson et al., 2018),[7] with LORELEI Resource Language Packs (LRLPs) in over 20 languages (Strassel and Tracey, 2016). Each LRLP contains several different layers of expert annotation, including 2 levels of named entity annotations (*simple*, with 4 tags, and *full*, which also include nominals and pronominals), parallel text, NP chunking, and POS tagging. The large scale of the project, and the consistency across languages make this an

[7] `https://www.darpa.mil/program/low-resource-languages-for-emergent-incidents`

attractive dataset for cross-lingual research.

Label Encodings

NER is typically cast as a sequence labeling task, in which each sentence is a sequence of words, and each word in a sentence is assigned a tag. In sequence labeling tasks, each element of the sequence is given exactly one tag. For POS tagging, this corresponds nicely with the stated goals: each word in context does indeed take exactly one POS tag. But in NER, name phrases often consist of a single token (‘Sting’), but frequently also consist of several tokens (‘Jascha Heifetz’, ‘e e cummings’). We need to encode this phrase-level information with token-level tags. One simplistic solution is to greedily concatenate all contiguous entity tags to form phrases. This would be fine if names with identical labels were never contiguous. But consider the following (contrived) example:

> The Cumberpatch Crutchley had known would never eat onions in the morning.

Clearly, “Cumberpatch” and “Crutchley” are two separate people, but a greedy tagging scheme would produce the name phrase “Cumberpatch Crutchley”. This example is contrived in English, but in languages in which more syntax is encoded in the morphology, it is common for name tokens from separate phrases to be contiguous.

As a result, there are a number of schemes for encoding phrases in sequential tags. The most common of these are IOB1, IOB2, and BILOU, first introduced in Ramshaw and Marcus (1995). These schemes work by adding prefixes to entity tags to make phrase boundaries. Each name refers to the possible prefixes prepended to the labels. Common to all schemes, the label for non-entities is ‘O’ for “Outside” or “Other.”

- **IOB1** This scheme prepends an “I-” (for Inside) to each name tag, and a ‘B-’ (for Begin) to any new contiguous name tag (e.g. Cumberpatch and Crutchley in the example above would have tags “I-PER B-PER”). The original CoNLL2003 English labels are encoded this way.

- **IOB2** This scheme is similar to IOB1, except the first token of each phrase (regardless of position or length) is always encoded with "B-".

- **BILOU** Also known as BIOUL (for Begin, Inside, Outside, Unit, Last), or BMOES (Begin, Middle, Outside, End, Single). "B-" and "I-" are the same as IOB2, but entities with only one token ("Sting") take "U-", and the last token of any phrase takes "L-".

All schemes are technically equivalent (they all encode exactly the same phrases) and can be transferred to each other deterministically with no loss.

The combination of encoding scheme (IOB1 or IOB2) with tagset {PER, ORG, LOC, MISC} leads to the actual labels used in the classifier. To calculate the actual number of labels used in the classifier:

$$\text{num tags} = \text{num non-O prefixes} \times \text{size of tagset} + 1$$

The +1 is for the O tag. For example, a tagset of {PER, ORG, LOC, MISC} used with the BILOU encoding yields $4 \times 4 + 1 = 17$ tags. Note that models treat B-PER and I-PER as completely distinct tags (they're just indices after all).

The tradeoffs of each tagging scheme are in the expressivity of the tagset compared with the number of tags. The BILOU scheme is more expressive, but also learns 4 different tags for each entity. Ratinov and Roth (2009) show a strong preference for the BILOU scheme, but Reimers and Gurevych (2017) suggests IOB2 outperforms BILOU.

Evaluation Metrics

Since most words are not named entities, it does not make sense to measure performance in terms of tag accuracy, as is done in part of speech tagging. A degenerate system that predicts 'O' for all tags would have accuracy of over 90% on most datasets. Instead, it is common to measure performance in terms of *phrase-based F_1 measure.* F_1 measure is the harmonic mean of precision and recall, which are defined as:

$$\text{precision} = \frac{\#\text{True Positives}}{\#\text{True Positives} + \#\text{False Positives}}$$

$$\text{recall} = \frac{\#\text{True Positives}}{\#\text{True Positives} + \#\text{False Negatives}}$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

To visualize these, it's helpful to think of an NER system as returning a set of name phrases from some target set of documents. Precision measures how many of the returned name phrases are correct. Recall measures how many of the total name phrases present in the document have been returned. F_1 is a harmonic mean of the two.

In certain situations, it may be desirable to target a system towards either precision or recall. For example, in Chapter 4.2 we instruct annotators to only annotate names if they have high confidence, which leads to high-precision low-recall annotations. In certain industry applications, it can be important to bias a system towards high precision (Arora et al., 2019).

Although it is less commonly reported, it is also possible to measure NER performance as *token-level* F_1. In this case, credit is given for each token that is labeled correctly, even if the full entity span is not correct. For example, suppose the test data has the name phrase "[Dariga Nursultanqyzy Nazarbayeva PER]", and the model predicts "[Dariga Nursultanqyzy PER] Nazarbayeva". On the phrase level, this counts as 1 false negative and 1 false positive.[8] On the token-level, this is 2 true positives, and 1 false negative. Because

[8]If it seems unfair to be penalized twice, see this blog post by Chris Manning for several interesting insights

of this partial credit, token-level F_1 is often higher than phrase-level F_1, so when comparing against prior work it is important to use the correct metric.

Some variations on evaluation exist. The Emerging and Rare entity recognition shared task at the Workshop for Noisy User Generated Text 2017 (WNUT) (Derczynski et al., 2017) measured performance only for unique surface strings. This reflects the fact that it's harder to find all of the different entities in a document than to find all instances of one surface string, such as "London". In the NER shared task at the Balto-Slavic Natural Language Processing 2019 workshop (Piskorski et al., 2019), performance is calculated in several different ways, including giving partial credit for detecting any morphological instantiation of a given person's name, or giving credit only for finding each unique variation of a name (similar to the unique surface measure of WNUT17).

2.2. Cross-lingual Methods

Having given a description of NER above, we now turn to related work in cross-lingual methods. There are two ways to categorize cross-lingual methods: according to algorithmic strategy, and according to resources used. We will first give a brief description of popular algorithmic strategies, but since our overarching goal is to understand cross-lingual methods in low-resource scenarios, we organize this survey according to available resources, ordered roughly from most expensive to least expensive.

When it comes to algorithmic strategy, prior work can be roughly divided into the categories of parallel projection, direct transfer, and the *create-and-clean* methods.

- **Parallel projection** Under this strategy, one computes word alignments (Brown et al., 1993) between each parallel sentence of a parallel corpus, and pushes annotations over the alignments from one side to the other. Then, with "annotated" target language data, a target language model can be trained. Assuming that source side

on evaluation in NER: `https://nlpers.blogspot.com/2006/08/doing-named-entity-recognition-dont.html` annotations are of high quality, success depends largely on the quality of the alignments, which depends, in turn, on the size of the parallel data, and the difficulty of aligning with the target language.

- **Direct transfer** Train a model using data in a high-resource language, and evaluate this model directly on the low-resource target language. This approach only works if the source and target languages are very similar, or if the model has used language-independent representations.

- **Create and clean** Use some noisy annotation process such as annotation rules or human non-speaker annotators to create noisy annotations directly in the target language, and apply automatic processes for cleaning these annotations. Ultimately, a model can be trained in the target language using "clean" annotations.

Now we turn to approaches with respect to resource levels. Naturally, these resource levels are not mutually exclusive, but represent common data scenarios. In certain cases, it can be wise to combine disparate resources (Mayhew et al., 2017b; Plank and Agić, 2018). We will focus mainly on NER, but include certain related tasks (such as POS or dependency tagging) as appropriate.

2.2.1. Methods using Parallel Text

Parallel text, in which sentences in one language are aligned with translations in another language, is a powerful cross-lingual resource. Although concepts can be expressed in different ways across different languages, it is often reasonable to assume that word alignments calculated automatically (Brown et al., 1993) can give in-context word-level translations. Much of the earliest work in cross-lingual methods used this resource, with examples in NP chunking (Yarowsky et al., 2001; Yarowsky and Ngai, 2001), POS tagging (Yarowsky and Ngai, 2001; Das and Petrov, 2011; Duong et al., 2014), NER (Wang and Manning, 2014; Kim et al., 2012; Ehrmann et al., 2011), and parsing (Hwa et al., 2005; Zeman and Resnik, 2008; Ganchev et al., 2009; McDonald et al., 2011). Because of noise in parallel data (Khayrallah and Koehn, 2018) or word alignments, projected annotations are often noisy, which can be addressed with constraints on the output (Yarowsky et al., 2001), or graphical models (Wang et al., 2013b).

McDonald et al. (2011) used parallel text as a weak constraint (Chang et al., 2007; Ganchev et al., 2010). Having first trained a delexicalized parser in English (or some other high-resource language), they predict parse trees over target language sentences, effectively creating a *silver standard* corpus, over which they train a lexicalized parser. Next, they iterate over a parallel corpus, using lexicalized parsers in English and the target language to make predictions on each sentence pair. They compare the parse tree on the English side with the top-k parses on the target side, and select the target parse tree that most closely matches the English tree.

Täckström et al. (2012) used parallel text to induce cross-lingual word clusters (i.e. Brown clusters (Brown et al., 1992)), which are then used as features in a direct transfer model. This idea of using parallel text to create cross-lingual representations was later visited in Luong et al. (2015) and Hermann and Blunsom (2014) for bilingual embeddings, and in Lample and Conneau (2019) for cross-lingual contextual embeddings.

Where most previous work does *hard projection*, in which labels are projected across alignments, Wang and Manning (2014) show that projecting expectations of labels can improve results. They experiment in two different settings: weakly-supervised, where only parallel data is available, and semi-supervised, where annotated training data is available along with unlabeled parallel data. Similar ideas were also found to be useful for POS (Agić et al., 2015, 2016).

But even when parallel text is available, there are different levels of quality. On the one hand, there are millions of lines of high-quality parallel text in a relatively general purpose domain, such as Europarl (Koehn, 2005). Such corpora are available for a relatively small number of languages. On the other hand, there are parallel corpora available for a much larger number of languages, but of questionable quality, such as the Bible (Christodouloupoulos and Steedman, 2015) and Watchtower (Agić and Vulić, 2019). In practice, these corpora are hard to work with (Enghoff et al., 2018), although there has been some work on POS taggers built from the Bible (Agić et al., 2015).

Most recently, Enghoff et al. (2018) suggests that cross-lingual projection for NER is not as simple as it seems, especially for low-resource languages, citing lack of parallel text of sufficient size or quality.

2.2.2. Methods using Wikipedia

Where the number of languages with high-quality parallel text may be limited, Wikipedia is available in around 300 languages. Wikipedia-based approaches exploit the fact that thousands of contributors have made annotations (edits) in hundreds of languages. Wikipedia has been used for a large number of NLP tasks, from use as a semantic space (Gabrilovich and Markovitch, 2007; Chang et al., 2008; Song and Roth, 2014), to generating parallel data (Smith et al., 2010), to use in open information extraction (Wu and Weld, 2010).

Wikipedia has also been used to extract training data for NER, under the intuition that it is already (partially) annotated with NER labels, in the form of links to pages. In one line of work (Nothman et al., 2008, 2013; Balasuriya et al., 2009; Nothman et al., 2009), *silver-standard* NER data is generated from Wikipedia using link targets and other heuristics. This can be gathered for any language in Wikipedia, but several of the heuristics depend on language-specific rules. Most Wikipedia pages contain *interwiki* links, which link a string in that page to some other Wikipedia page. For example, a page about "Pizza" may have a phrase describing "[New York] style", and "New York" is a hyperlink to the Wikipedia page for "New York City." In fact, predicting this link structure is the goal of a separate task, sometimes called wikification (Ratinov and Roth, 2011; Tsai and Roth, 2016b). In this example, the phrase "New York" corresponds to a location entity. Another important step in this task is to classify Wikipedia pages with respect to named entity types (Nothman et al., 2013). These procedures were refined in Pan et al. (2017), with the addition of some extra features for page classification (leading to a small increase in classification performance), and using self-training instead of surface propagation. Al-Rfou et al. (2015) generate training data from Wikipedia articles using a similar manner.

Kim et al. (2012) use Wikipedia to generate parallel sentences with NE annotations, which they then project across, as described in the previous section. Kazama and Torisawa (2007) do NER using Wikipedia category features for each mention. However, their method for wikifying text is not robust to ambiguity, and they only do monolingual NER.

Sil and Yates (2013) create a joint model for NER and entity linking in English. They avoid the traditional pipeline by overgenerating mentions in the first stage and using NER features to rank candidates. While the results are promising, the model is not scalable to other languages because it requires both a trained NER and a NP chunker.

Wikipedia has also been used to train cross-lingual representations. In particular, BERT (Devlin et al., 2019) trained over the top 100 languages from Wikipedia has shown extraordinary cross-lingual properties despite having no cross-lingual training objectives (Pires et al., 2019; Wu and Dredze, 2019).

In Chapter 4.1, we describe a method for generating language-independent representations from Wikipedia inter-language links, as presented in Tsai et al. (2016).

2.2.3. Methods using Lexicons

While Wikipedia is useful as a multilingual resource, it's usefulness decreases with the size of the Wikipedia. For example, the Oromo Wikipedia has only 787 pages,[9] and very few links, either interlanguage or interwiki, and as a result is almost indistinguishable from a small monolingual corpus. For those languages that lack parallel text, and the rich multilingual structure of Wikipedia, it is often possible to find lexicons, for example, from PanLex (Kamholz et al., 2014). These provide word-level cross-lingual supervision, which despite

[9]`https://en.wikipedia.org/wiki/List_of_Wikipedias`, accessed August 7, 2019

lacking the contextual aspect of word alignments in parallel text, can still provide useful cross-lingual signals.

Wick et al. (2016) show that if a word embedding model is tasked with predicting a *word translation* given the context, then the resulting embeddings exhibit cross-lingual properties. To create this data, they use a lexicon to translate random words in a corpus, and train a standard word embedding model over this "artificially code-switched" data.

For many applications, it can be beneficial to train cross-lingual embeddings. Faruqui and Dyer (2014) use lexicons as anchor points to tie together two monolingual vector spaces, using Canonical Correlation Analysis (CCA). Similar methods were also used by Conneau et al. (2018) and Wick et al. (2016).

In Section 4.2, we describe a method for translating text in a high-resource language word for word into a low-resource language given only a lexicon (Mayhew et al., 2017b). This work was extended by (Xie et al., 2018), which induced dictionaries from cross-lingual word embeddings, and added a self-attention layer.

Similar methods have been used for cross-lingual document classification (Song et al., 2016, 2019), cross-lingual information retrieval (Ballesteros and Croft, 1996; Hull and Grefenstette, 1996), and opinion target extraction (Liu et al., 2012).

2.2.4. Methods using Monolingual Text

In recent years, some ambitious work has attempted to create cross-lingual word embeddings using nothing but monolingual text (Conneau et al., 2018; Artetxe et al., 2018). The basic assumption is that all monolingual embedding spaces are essentially isomorphic, allowing separate spaces (say embeddings in English and Russian) to be aligned simply by finding the correct rotation matrix. One way to discover this matrix is through an adversarial approach that prevents a discriminator from distinguishing embeddings randomly sampled from either space. Similarly, Artetxe et al. (2018) hypothesize that word similarity distributions are similar across languages. Another way to discover this matrix is to use shared tokens as anchors. For similar languages (e.g. English and Spanish) this may include shared words, cognates, named entities, and punctuation and number tokens. But for languages that are not similar (especially with separate scripts), there may be no identical tokens.

2.2.5. Methods using Human Annotators

While nearly every NLP corpus is created with human annotators that speak the corpus language (usually English), it may also be possible to use humans to annotate text they do not speak. There has been little work in NLP in this area, but some in related fields. Most notably the concept of "mismatched annotation" in speech recognition (Jyothi and Hasegawa-Johnson, 2015a,b; Do et al., 2018), in which a person listens to spoken text in an unfamiliar language and writes down, using English orthography, the syllables that most closely match what they hear. The result is usually nonsense, but with redundancy and specialized decoding methods, a reasonable target language transcription emerges.

Participants in the LORELEI evaluations have found that non-speaker annotations have been very valuable (Mayhew et al., 2019b), especially when facilitated with tools such as TALEN (Mayhew and Roth, 2018) and others (Lin et al., 2018).

In Section 3.2, we show a method for training a high-quality classifier from high-precision low-recall training data, with a non-speaker annotator experiment at the end. In Section 3.1, we show experiments quantifying the quality of non-speaker annotators as compared to native speakers of a target language. In our final chapter, Chapter 6, we outline a process for developing an NER system in a surprise language in a short time frame. A large part of this process is using non-speaker annotations.

CHAPTER 3 : Indirect Signals for NER

3.1. Non-Speaker Annotations: Tools and Experiments

3.1.1. TALEN: Tool for Annotating Low-resource ENtities

This work published as (Mayhew and Roth, 2018).

Introduction

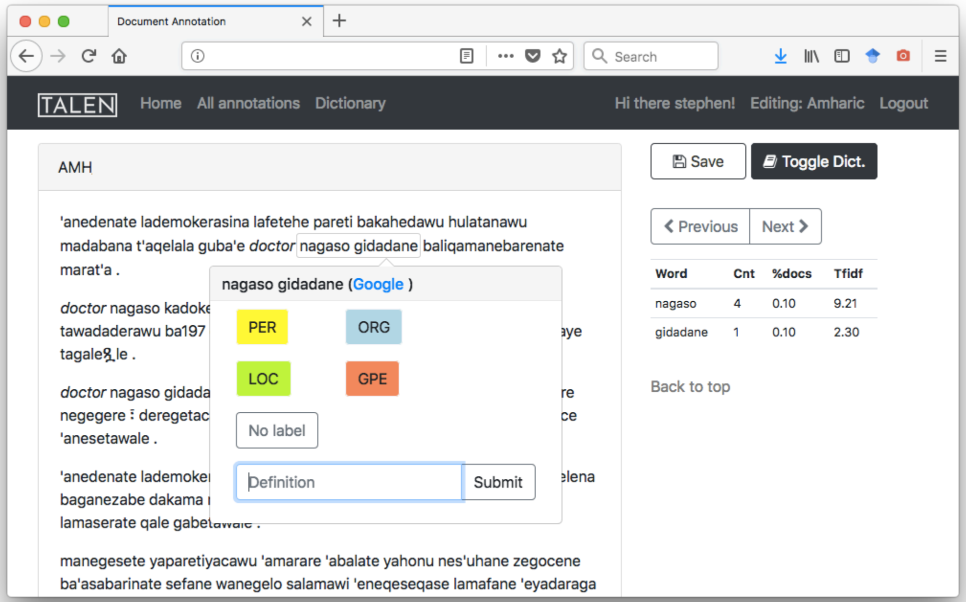
Language annotation strategies and software have historically assumed that annotators speak the language in question. Although there has been work on *non-expert* annotators for natural language tasks (Snow et al., 2008), where the annotators lack specific skills related to the desired task, there has been little to no work on situations where annotators, expert or not, do not speak the target language. To this end, we present a web-based interface designed for users to annotate text quickly and easily in a language they do not speak.

This problem has risen on the same tide as interest in low-resource research, where it can be difficult to find annotators who speak the target language. Cross-lingual techniques (Mayhew et al., 2017b) have been proposed for low-resource scenarios, but often the first step is to present romanized foreign language text to English speakers, and hope for the best.

Talen aids non-speaker annotators[1] with several different helps and nudges that would be unnecessary in cases of a native speaker. The main features, described in detail in Section 3.1.1, are an NER specific interface, entity propagation, lexicon integration, token statistics information, and internet search.

The tool operates in two separate modes, each with all the helps described above. The

[1]We use 'non-speaker' to denote a person who does not speak or understand the language, in contrast with 'non-native speaker', which implies at least a shallow understanding of the language.



AMH

'anedenate lademokerasina lafetehe pareti bakahedawu hulatanawu madabana t'aqelala guba'e *doctor* nagaso gidadane baliqamanebarenate marat'a .

doctor nagaso kadoke … tawadaderawu ba197 … aye tagaleጺle .

doctor nagaso gidada … re negegere ፧ deregetac … ce 'anesetawale .

'anedenate lademoke … elena baganezabe dakama … lamaserate qale gabetawale .

manegesete yaparetiyacawu 'amarare 'abalate yahonu nes'uhane zegocene ba'asabarinate sefane wanegelo salamawi 'eneqeseqase lamafane 'eyadaraga

Word	Cnt	%docs	Tfidf
nagaso	4	0.10	9.21
gidadane	1	0.10	2.30

Figure 3: Document-based annotation screen. A romanized document from an Amharic corpus is shown. The dictionary is active, and is displaying the definition (in italics) for "doctor". The user has selected 'nagaso gidadane' (Negasso Gidada) for tagging. (URL and document title are obscured).

first mode displays atomic documents in a manner analogous to nearly all prior annotation software. The second mode operates on the sentence level, patterned on bootstrapping with a human in the loop, and designed for efficient discovery and annotation.

In addition to being useful for non-speaker annotations, the tool can be used as a lightweight inspection and annotation tool for NER, or any other sequence tagging task. In practice, because of the compact display, the tool works best for sparse annotations, such as NE, and is less useful for dense annotations, such as POS.

Main Features

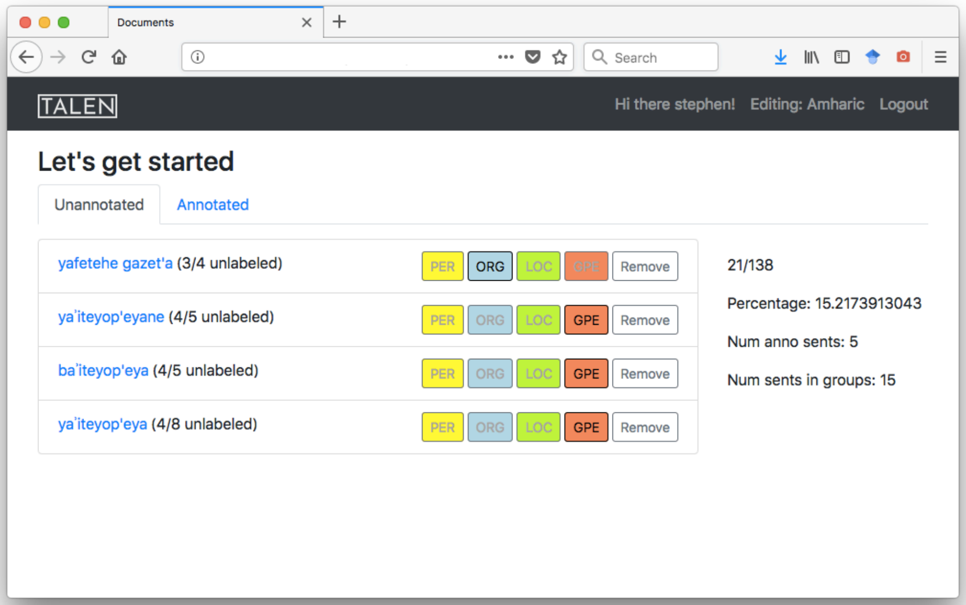
In this section, we describe the main features individually in detail.

NE specific interface The interface is designed specifically for NE annotation, where entities are relatively rare in a document. The text is intentionally displayed organically, in a way that is familiar and compact, so that the annotator can see as much as possible in any given screen. This makes it easy for an annotator to make document-level decisions, for example, if an unusual-looking phrase appears several times. To add an annotation, as shown in Figure 3, the annotator clicks (and drags) on a word (or phrase), and a popover appears with buttons corresponding to label choices as defined in the configuration file. Most words are not names, so a default label of non-name (O) is assigned to all untouched tokens, keeping the number of clicks to a minimum.

In contrast, a part-of-speech (POS) annotation system, SAWT (Samih et al., 2016), for example, is designed so that every token requires a decision and a click. For NE annotation, it is reasonable to assume that any token that has not been annotated can take a default label of non-name.

Entity Propagation In an effort to save clicks, the interface propagates all annotation decisions to every matching surface in the document. For example, if *'iteyop'eya* (Ethiopia) shows up many times in a document, then a single click will annotate all of them. In a low-resource scenario, it can be difficult to discover and notice names because all tokens look unfamiliar. This mechanism eases some of this burden.

Lexicon Integration Naturally, the difficulty for non-speakers is that they do not understand the text they are annotating. An important feature of TALEN is in-place lexicon integration. This functionality replaces words in the annotation screen with their translation from the lexicon. As before, this is built on the notion that annotation is easiest when text looks organic, in this case, when the translations are inline with the foreign text. If a bilingual lexicon is available, this can be used. If no lexicon is available, the user can click on a token and add definitions, which are immediately saved to disk for future use. Now, the next time the annotator encounters this word, the definition will be displayed. A side



Let's get started

Unannotated | Annotated

yafetehe gazet'a (3/4 unlabeled)	PER	ORG	LOC	GPE	Remove	21/138
ya'iteyop'eyane (4/5 unlabeled)	PER	ORG	LOC	GPE	Remove	Percentage: 15.2173913043
ba'iteyop'eya (4/5 unlabeled)	PER	ORG	LOC	GPE	Remove	Num anno sents: 5
ya'iteyop'eya (4/8 unlabeled)	PER	ORG	LOC	GPE	Remove	Num sents in groups: 15

Figure 4: Sentence-based annotation screen showing 4 seed terms available for annotation. Notice the *Unannotated* and *Annotated* tabs. These terms are in the active *Unannotated* tab because each term has some sentences that have not yet been labeled with that seed term. For example, of the 5 sentences found for *ba'iteyop'eya*, only 1 has this seed term labeled (see Figure 5).

effect of this feature is that a lexicon will be collected, and can be shared with other users, or used for other tasks.

Token Statistics When one has no knowledge of a language, it can be useful to know various statistics about tokens, such as document count, corpus count, or TF-IDF. Our annotation screen shows a table with statistics over tokens, including document count, percentage of documents it is present in, and TF-IDF. At first, it shows the top 10 tokens by TF-IDF, but the user can click on any token to get individual statistics. In practice, we have found that this helps to give an idea of the topic of the document, and often names will have high TF-IDF in a document.

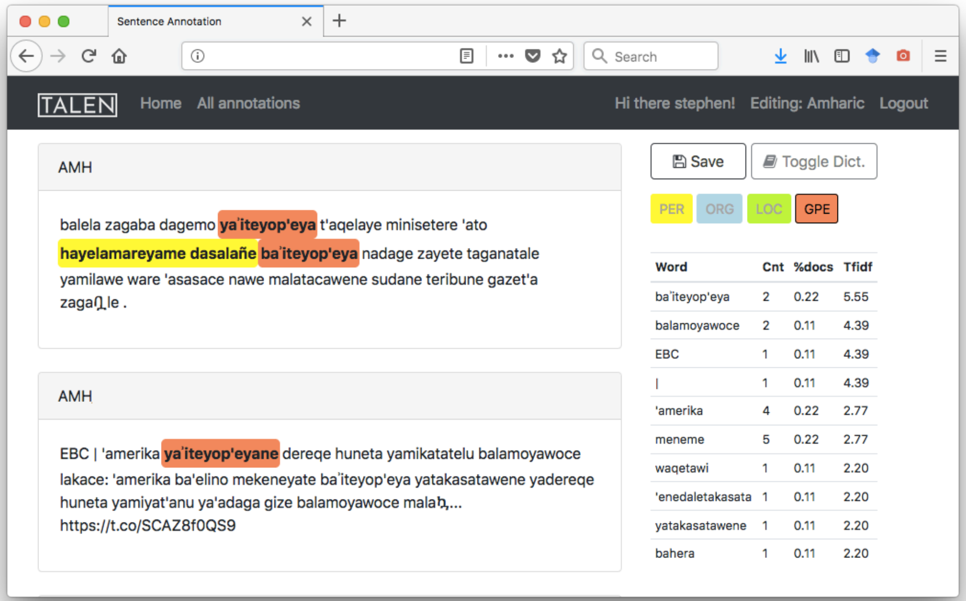
Internet Search Upon selection of any token or phrase, the popover includes an external link to search Google for that phrase. This can be helpful in deciding if a phrase is a name or not, as search results may return images, or even autocorrect the phrase to an English standard spelling.

Annotation Modes

There are two annotation modes: a document-based mode, and a sentence-based mode. Each has all the helps described above, but they display document and sentences to users in different ways.

Document-based Annotation The document-based annotation is identical to the common paradigm of document annotation: the administrator provides a group of documents, and creates a configuration file for that set. The annotator views one document at a time, and moves on to the next when they are satisfied with the annotations. Annotation proceeds in a linear fashion, although annotators may revisit old documents to fix earlier mistakes. Figure 3 shows an example of usage in the document-based annotation mode. This is likely to be the most popular usage of this tool because of its simplicity and familiarity.

Sentence-based Annotation The sentence-based annotation mode is modeled after bootstrapping methods. In this mode, the configuration file is given a path to a corpus of documents (usually a very large corpus) and some expected seed entities. This corpus is indexed at the sentence level, and k sentences containing the seed entities are retrieved. These are presented to the annotator, as in Figure 5, who will mark all names in the sentences, starting with the entity used to gather the sentence, and hopefully discovering other names in the process. As names are discovered, they are added to the list of seed entities, as shown in Figure 4 New sentences are then retrieved, and the process continues until a predefined number of sentences has been retrieved. At this point, the data set is frozen, no new sentences are added, and the annotator is expected to thoroughly examine each sentence to discover and annotate all named entities.



AMH

balela zagaba dagemo **ya'iteyop'eya** t'aqelaye minisetere 'ato **hayelamareyame dasalañe** **ba'iteyop'eya** nadage zayete taganatale yamilawe ware 'asasace nawe malatacawene sudane teribune gazet'a zagaበle .

AMH

EBC | 'amerika **ya'iteyop'eyane** dereqe huneta yamikatatelu balamoyawoce lakace: 'amerika ba'elino mekeneyate ba'iteyop'eya yatakasatawene yadereqe huneta yamiyat'anu ya'adaga gize balamoyawoce malaክ,... https://t.co/SCAZ8f0QS9

Word	Cnt	%docs	Tfidf
ba'iteyop'eya	2	0.22	5.55
balamoyawoce	2	0.11	4.39
EBC	1	0.11	4.39
|	1	0.11	4.39
'amerika	4	0.22	2.77
meneme	5	0.22	2.77
waqetawi	1	0.11	2.20
'enedaletakasata	1	0.11	2.20
yatakasatawene	1	0.11	2.20
bahera	1	0.11	2.20

Figure 5: Sentence-based annotation screen, with sentences corresponding to seed term *ba'iteyop'eya* (annotated in the first sentence, not in the second). Two sentences are shown, and the remaining three sentences associated with this seed term are lower on the page. Notice that *hayelamareyame dasalañe* (Hailemariam Desalegn) has also been annotated in the first sentence. This will become a new seed term for future iterations.

In practice, we found that the size of k, which is the number of sentences retrieved per seed term, affects the overall corpus diversity. If k is large relative to the desired number of sentences, then the annotation is fast (because entity propagation can annotate all sentences with one click), but the method produces a smaller number of unique entities. However, if k is small, annotation may be slower, but return more diverse entities. In practice, we used a $k = 5$.

Experiment: Compare to brat

The brat rapid annotation tool (brat) (Stenetorp et al., 2012) is a popular and well-featured annotation tool, which makes for a natural comparison to TALEN. In this experiment, we compare tools qualitatively and quantitatively by hiring a group of annotators. We can compare performance between Talen and brat by measuring the results after having annotators use both tools.

We chose to annotate Amharic, a language from Ethiopia. We have gold training and test data for this language from the LORELEI project. The corpus is composed of several different genres, including newswire, discussion forums, web blogs, and social network (Twitter). In the interest of controlling for the domain, we chose only the 125 newswire documents (NW) from the training data, and removed all annotations before distribution. Since Amharic is written in Ge'ez script, we romanized it, so it can be read by English speakers. We partitioned the newswire documents into 12 (uneven) groups of documents, and assigned each annotator 2 groups: one to be annotated in brat, the other with Talen. This way, every annotator will use both interfaces, and every document will be annotated by both interfaces. We chose one fully annotated gold document and copied it into each group, so that the annotators have a sample of what to expect.

We employed 12 annotators chosen from our NLP research group. Before the annotation period, all participants were given a survey about tool usage and language fluency. No users had familiarity with Talen, and only one had any familiarity with brat. Of the annotators, none spoke Amharic or any related language, although one annotator had some familiarity with Hebrew, which shares a common ancestry with Amharic, and one annotator was from West Africa.[2]

Immediately prior to the annotation period, we gave a 15 minute presentation with instructions on tool usage, annotation guidelines, and annotation strategies. The tags used were Person, Organization, Location, and Geo-political entity, As for strategy, we instructed them to move quickly, annotating names only if they are confident (e.g. if they know the English version of that name), and to prioritize diversity of discovered surface forms over exhaustiveness of annotation. When using Talen, we encouraged them to make heavy use of the dictionary. We provided a short list (less than 20 names) of English names that are

[2]Ethiopia is in East Africa.

likely to be found in documents from Ethiopia: local politicians, cities in the region, etc.

The annotation period lasted 1 hour, and consisted of two half hour sessions. For the first session, we randomly assigned half the annotators to use brat, and the other half to use Talen. When this 30 minute period was over, all annotators switched tools. Those who had used brat use ours, and vice versa. We did this because users are likely to get better at annotating over time, so the second tool presented should give better results. Our switching procedure mitigates this effect.

At the end of the second session, each document group had been annotated twice: once by some annotator using brat, and once by some annotator using Talen. These annotations were entirely separate, so each tool started with a fresh copy of the data.

We report results in two ways: first, annotation quality as measured against a gold standard, and second, annotator feedback.

Figure 6 shows basic statistics on the datasets. Since the documents we gave to the annotators came from a gold annotated set, we calculated precision, recall, and F1 with respect to the gold labels. First, we see that Talen gives a 5.8 point F1 improvement over brat. This comes mostly from the recall, which improves by 3.9 points. This may be due to the automatic propagation, or it may be that having a dictionary helped users discover more names by proximity to known translations like *president*. In a less time-constrained environment, users of brat might be more likely select and annotate all surfaces of a name, but the reality is that all annotation projects are time-constrained, and any little help is valuable.

The bottom part of the table shows the annotation statistics from Talen compared with brat. Talen yielded about 50 more name annotations than brat, which is also likely a product of the name propagation. The number of unique names further confirms this: Talen returns more names overall, but brat returns more unique names.

We gathered qualitative results from a feedback form filled out by each annotator after

Dataset	Precision	Recall	F1
brat	51.4	8.7	14.2
Talen	53.6	12.6	20.0

Dataset	Total names	Unique names
Gold	2260	1154
brat	405	189
Talen	457	174

Figure 6: Performance results. The precision, recall, and F1 are measured against the gold standard Amharic training data.

the evaluation. All but one of the annotators preferred Talen for this task. In another question, they were asked to select an option for 3 qualities of each tool: efficiency, ease of use, and presentation. Each quality could take the options *Bad, Neutral,* or *Good.* On each of these qualities, brat had a majority of *Neutral*, and Talen had a majority of *Good.* For Talen, *Efficiency* was the highest rated quality, with 10 respondents choosing *Good.*

We also presented respondents with the 4 major features of Talen (TF-IDF box, dictionary, entity propagation, Google search), and asked them to rate them as *Useful* or *Not useful* in their experience. Only 4 people found the TF-IDF box useful; 10 people found the dictionary useful; all 12 people found the entity propagation useful; 7 people found the Google search useful. These results are also reflected in the free text feedback. Most respondents were favorable towards the lexicon, and some respondents wrote that the TF-IDF box would be useful with more exposure, or with better integration (e.g. highlight on hover).

3.1.2. Quantifying Non-Speaker Quality

This work done in collaboration with Tatiana Tsygankova.

In our experience with building practical low-resource systems (see Chapter 6), we discovered that one remarkably powerful tool is human annotators who don't speak the target language. We refer to such annotators as non-speaker (NS) annotators, to differentiate them from non-native speakers, who may still speak the language. Although this technique has seen a small amount of work in NER (Mayhew and Roth, 2018; Lin et al., 2018), machine translation (Hermjakob et al., 2018b), and low-resource speech recognition (Jyothi and Hasegawa-Johnson, 2015a,b), this idea that non-speakers can produce good annotations is still largely unknown and unquantified in the broader community.

In this work, we aim to build a better understanding of this process, asking questions about the quality of non-speaker annotations in an absolute sense, and also in relation to annotations from a native speaker of the language, which we refer to as native informant (NI) for historical reasons. In particular, our experimental questions are:

1. Can NS annotators actually produce meaningful annotations?
2. How to compare NS/NI annotations on a fixed budget of annotation time?
3. How do NS/NI annotations change over time?
4. How best to combine NS/NI annotations?

Experimental Setup In order to test these questions, we set up an annotation exercise using NS and NI annotators. We chose to use Russian, because of availability of annotators and data. All annotation was done using TALEN on LRLP data with gold annotations removed. In particular, the data given to annotators comes from the train split. By using gold annotated data, we can measure performance in two ways: 1) directly against the gold annotations and 2) by training a model on the annotated text and evaluating on the test split. We used Russian annotated data (LDC2016E95) from the LORELEI project, with our own train/test split.

Annotations were done over 4 sessions, each lasting 45 minutes. There was a 15 minute break between the first two sessions, a lunch break in the middle, and then two more consecutive sessions separated by a 15 minute break. Not all annotators participated in every session, but for the sake of the third research question, we ensured that the NI and three NS annotators committed to participate in the full schedule.

We used a total of 9 NS annotators, composed of graduate and undergraduate students from the University of Pennsylvania and some other institutions. Languages represented among

	NI	NS
Annotation Time	3 hrs	15 hrs
Dataset size (tokens)	19K	36K
Annotation quality (F1)	76.5	47.8
Model performance (F1)	55.6	49.6

Table 1: Overall performance of the annotation experiment aggregated over all sessions. The 3 hours of NI annotation time are from 4 sessions of 45 minutes each. The 15 hours of NS annotation time are from 20 sessions of 45 minutes each (12 sessions from 3 committed annotators, and 8 sessions from the non-committed annotators).

annotators were English, Chinese, Korean, and Shona. In preparation for the annotation task, annotators were asked beforehand to study the annotation guidelines and complete 30 minutes of annotation over English data. This gave annotators practice with the interface, and allowed us to give feedback on guideline misunderstandings. Annotators were compensated for their time with a free lunch. We had one NI, a native speaker of Russian, who is familiar with the task, and also an author of this work.

We prepared the data given to the committed annotators in the following way. We took a set of documents and split it into three equally sized subsets. The NI was given the entire set, under the hypothesis that an NI would work faster, and could potentially get through more documents than an NS. We assigned the subsets to the three committed NS annotators, so that they could work on them in parallel. We chose to give the NI and NS annotators the same set of documents so that we could directly compare performance. All data given to the NS annotators was romanized. At the end of each annotation session, we took a "snapshot" of each annotator's data annotated up to that point. Having these snapshots would allow us to make observations based on cumulative time spent annotating, and track performance over time.

For the non-committed annotators, we split a separate set of documents into 20 subsets. At each session, these annotators selected a subset for annotation, thus avoiding duplicate annotations.

Results We present overall results in Table 1, showing performance of the NI compared to all NS annotators (committed and non-committed) over an equal period of wall-clock annotation time. To be precise, the 3 hours of NI annotation time are from 4 sessions of 45 minutes each. The 15 hours of NS annotation time are from 20 sessions of 45 minutes each (12 sessions from 3 committed annotators, and 8 sessions from the non-committed annotators).

First, we see that NS annotators are able to annotate nearly twice as many tokens in aggregate than the NI, but per person they annotated on average about 4K tokens, much less than the NI. This is evidence in favor of our hypothesis that NIs can annotate faster than NSs. Next we can examine the performance both in terms of direct evaluation, and in terms of model evaluation.

Evaluating directly, we see that the NI performance is 76.5, compared to NS performance of 47.8. This is as we might expect: the NS annotations are poor quality. However, when we train a model on each set of annotations and evaluate on a held-out test set, the performance gap shrinks to only 6 points F1. This shows that even though the NS annotations are demonstrably low quality, the large size of the annotation set mostly compensates for this.

For all following experiments, we examine only the annotations of the committed NI and NS annotators.

Next, we examine quality of annotations as measured directly against the gold standard over the 4 annotation sessions. Since all data given to the annotators already had gold annotators, we can measure

In this measurement, the size of the data makes no difference to the outcome (except that the outcome may have more statistical significance). Figure 7 shows the relevant figure. The three columns represent NI (blue), average of all 3 NSs (red), and worst single NS (yellow).

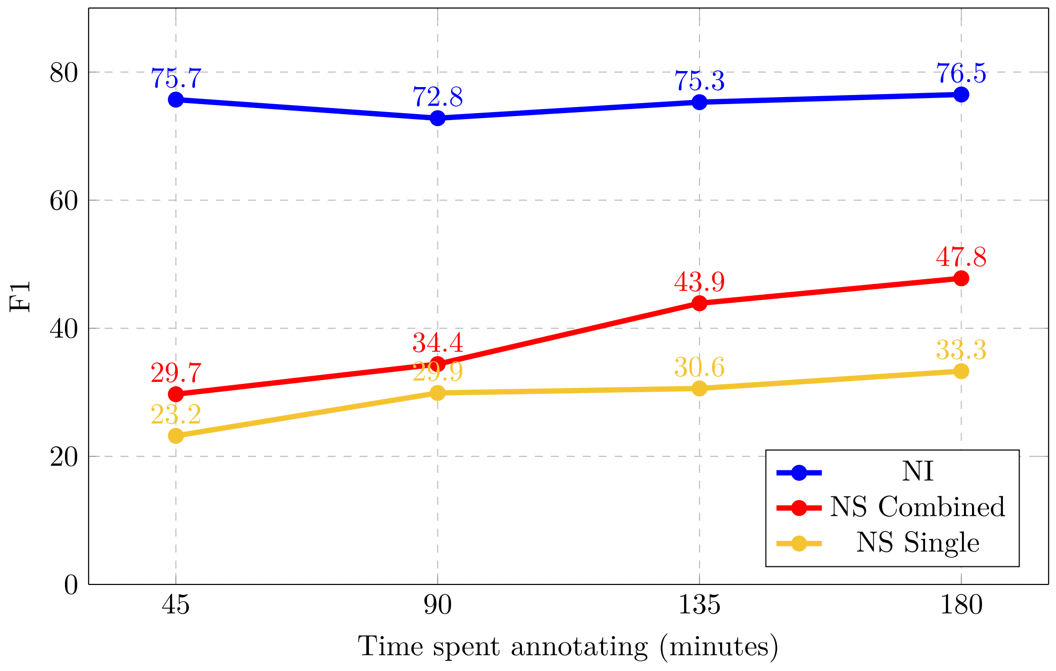


Figure 7: Directly quantifying NS annotations against gold annotations over time. Each score is F1, and is calculated using gold annotations as reference. The blue line represents NI annotations. The red line represents the average of the 3 NS annotators. The yellow line represents the lowest-performing single NS annotator.

First, NI performance is consistently high, and doesn't change much over time. This is consistent with expectations, in that we wouldn't expect a native speaker to improve in language skills. However, the NS annotation quality rises significantly, both on average and the single worst annotator. This suggests that NS annotators learn relevant aspects of the language as they annotate, even over 3 hours of annotating.

Next we use the annotations as training data for some model, and measure performance on a held out test set. We used a standard BiLSTM-CRF model (Ma and Hovy, 2016) from AllenNLP (Gardner et al., 2018) with fastText 300 dimensional embeddings (Bojanowski et al., 2017). Results are shown in Figure 8. In this figure, the blue trend line shows at each point the performance of a model trained on all NI data annotated up to that point. The red trend line is similar, except it is the aggregate of all NS data up to that point (which may be significantly larger than the NI point at the corresponding time). The yellow line shows the average performance of the three NS models having been trained separately at each point.

As before, we see a consistent trend of improvement, even from the single models. Also as before, we see that combining weak annotations (yellow line) gives a significant boost (red line), although not enough to reach the NI performance (blue line). However, we may ask, how much time of NS annotation is required to match (or beat) NI annotation scores? To answer this, we look at the NS performance at the 180 minute time point (after session 4) compared to the NI performance at the 90 minute time point (after session 2). Respectively, these are 45.8 and 43.7, which are roughly comparable. Since there are 3 NS annotators, and the time points differ by a factor of two, this suggests a new rule of thumb: one hour of NI annotation is worth roughly 6 hours of NS annotation. Naturally, this comes with many caveats: it is just one experiment, with one small set of annotators, on one (arguably) easy language. However, we may treat it as a first order estimation.

One may argue that the NS performance of 45.8 is hardly a strong score for any NER task. However, this is the outcome of a relatively small amount of annotation (12 person-hours), and the trend lines suggest that more time would give improved performance.

Finally, we address the question of improving NI performance by pre-annotating. Specifically, if we assume limited interaction time with the NI, how can we make the best of it? Our proposed method is to annotate text with NS annotators before giving it to the NI. Even if the NS is unable to exhaustively annotate, perhaps they can find "easy" entities, and save the NI some clicks. Then the NI can focus on annotations that are too difficult for an NS annotator.

To test this, we looked at the performance of the NI after one 45-minute session of annotating blank documents. This is the first point in Figure 8. Then, we took documents annotated by the non-committed NS annotators (6 sessions worth, roughly 4.5 person hours of annotation), and presented them to the NI for an additional hour of annotation. Re-

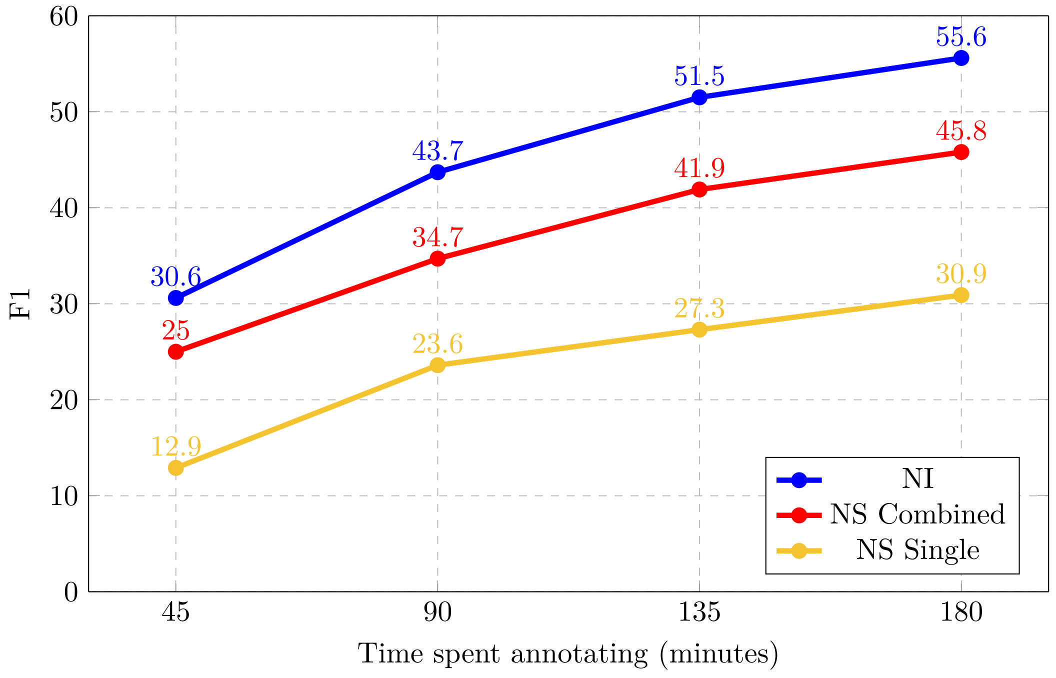


Figure 8: Using annotations as training data and evaluating on a held out Russian test set. The blue line represents scores from a model trained on NI annotations at each time point. The red line represents scores from a model trained on all committed NS data annotations at each time point. The yellow line shows the average of scores from models trained on NS annotations separately.

	NI (from scratch)	NS $\rightarrow$ NI
Annotation time	1 hr	(4.5 hrs) $\rightarrow$ 1 hr
Dataset size (tokens)	4K	9K
Annotation quality (F1)	75.7	71.0
Model performance (F1)	30.6	44.3

Table 2: How best to combine NI and NS.

call that the documents given to the non-committed NS annotators are distinct from those originally given to the NI, thus ensuring that the NI has not seen them before. In this way, we can compare the annotation efficiency of the NI annotating from scratch, or annotating from preannotated documents.

The results are shown in Table 2, in which the two described scenarios correspond to columns of the table. Most importantly, we see that from scratch, the NI annotates about 4K tokens, but manages to annotate 9K tokens from the preannotated set. The next two rows show that while the direct evaluation quality may have suffered slightly (perhaps because of a bias from the NS annotators), the model performance has taken a significant boost. As another rule of thumb, based on the number of tokens annotated, we can say that this preannotation procedure doubles NI efficiency. As above, many caveats apply, but this can be treated as a first order approximation.

3.1.3. Conclusions

This has been the first quantitative study of non-speaker annotators for NER. We have shown that NS annotators are capable of producing meaningful (if errorful) annotations, and that more annotations are always better, even despite questionable quality. As a rule of thumb, we argue that 1 hour of NI annotation time is worth 6 hours of NS time, and that NS preannotated text causes NI efficiency to double.

However, an important caveat is that all these results are with respect to a certain set of annotators (who may be unusually high quality), using Russian language data (which may be easier than other languages), in a relatively short time frame. While these results are useful for establishing estimates, we look forward to future research that examines this framework on a larger scale.

3.2. Named Entity Recognition with Partial Annotation

This work published as (Mayhew et al., 2019a)

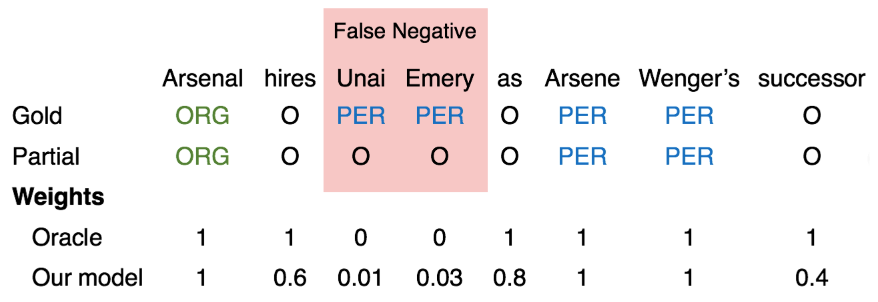
3.2.1. Introduction

Most modern approaches to NLP tasks rely on supervised learning algorithms to learn and generalize from labeled training data. While this has proven successful in high-resource scenarios, this is not realistic in many cases, such as low-resource languages, as the required amount of training data just doesn't exist. However, partial annotations are often easy to gather.

We study the problem of using partial annotations to train a Named Entity Recognition (NER) system. In this setting, all (or most) identified entities are correct, but not all entities have been identified, and crucially, there are no reliable examples of the negative class. The sentence shown in Figure 9 shows examples of both a gold and a partially annotated sentence. Such partially annotated data is relatively easy to obtain: for example, a human annotator who does not speak the target language may recognize common entities, but not uncommon ones. With no reliable examples of the negative class, the problem becomes one of estimating which unlabeled instances are true negatives and which are false negatives.

To address the above-mentioned challenge, we present Constrained Binary Learning (CBL) – a novel self-training based algorithm that focuses on iteratively identifying true negatives for the NER task while improving its learning. Towards this end, CBL uses constraints that incorporate background knowledge required for the entity recognition task.

We evaluate the proposed methods in 8 languages, showing a significant ability to learn from partial data. We additionally demonstrate the improvement that can be achieved by initializing CBL with domain-specific instance-weighing schemes. In the process, we use



			False Negative					
	Arsenal	hires	Unai	Emery	as	Arsene	Wenger’s	successor
Gold	ORG	O	PER	PER	O	PER	PER	O
Partial	ORG	O	O	O	O	PER	PER	O
Weights								
Oracle	1	1	0	0	1	1	1	1
Our model	1	0.6	0.01	0.03	0.8	1	1	0.4

Figure 9: This example has three entities: *Arsenal*, *Unai Emery*, and *Arsene Wenger*. In the *Partial* row, the situation addressed in this paper, only the first and last are tagged, and all other tokens are assumed to be non-entities, making *Unai Emery* a false negative as compared to *Gold*. Our model is an iteratively learned binary classifier used to assign weights to each token indicating its chances of being correctly labeled. The *Oracle* row shows optimal weights.

weighted variants of popular NER models, showing strong performance in both non-neural and neural settings. Finally, we show experiments in a real-world setting, by employing non-speakers to manually annotate romanized Bengali text. We show that a small amount of non-speaker annotation combined with our method can outperform previous methods.

3.2.2. Related Work

The supervision paradigm in this paper, partial supervision, falls broadly under the category of semi-supervision (Chapelle et al., 2009), and is closely related to weak supervision (Hernández-González et al., 2016)[3] and incidental supervision (Roth, 2017), in the sense that data is constructed through some noisy process. However, all of the most related work shares a key difference from ours: reliance on a small amount of fully annotated data in addition to the noisy data.

Fernandes and Brefeld (2011) introduces a transductive version of structured perceptron for partially annotated sequences. However, their definition of partial annotation is labels removed at random, so examples from all classes are still available if not contiguous.

Fidelity Weighted Learning (Dehghani et al., 2017) uses a teacher/student model, in which

[3]See also: `https://hazyresearch.github.io/snorkel/blog/ws_blog_post.html`

the teacher has access to (a small amount) of high quality data, and uses this to guide the student, which has access to (a large amount) of weak data.

Hedderich and Klakow (2018), following Goldberger and Ben-Reuven (2017), add a noise adaptation layer on top of an LSTM, which learns how to correct noisy labels, given a small amount of training data.

In the world of weak supervision, Snorkel (Ratner et al., 2017; Fries et al., 2017), is a system that combines automatic labeling functions with data integration and noise reduction methods to rapidly build large datasets. They rely on high recall and consequent redundancy of the labeling functions. We argue that in certain realistic cases, high-recall candidate identification is unavailable.

We draw inspiration from the Positive-Unlabeled (PU) learning framework (Liu et al., 2002, 2003; Lee and Liu, 2003; Elkan and Noto, 2008). Originally introduced for document classification, PU learning addresses problems where examples of a single class (for example, sports) are easy to obtain, but a full labeling of all other classes is prohibitively expensive. In our experiments, we compare with the class-weighted SVM of Liu et al. (2003).

Named entity classification as an instance of PU learning was introduced in Grave (2014), which uses constrained optimization with constraints similar to ours. However, they only address the problem of named entity classification, in which mentions are given, and the goal is to assign a *type* to a named-entity (like ‘location’, ‘person’, etc.) as opposed to our goal of identifying and typing named entities.

Although the task is slightly different, there has been work on building ‘silver standard’ data from Wikipedia (Nothman et al., 2008, 2013; Pan et al., 2017), using hyperlink annotations as the seed set and propagating throughout the document.

Partial annotation in various forms has also been studied in the contexts of POS-tagging (Mori et al., 2015), word sense disambiguation (Hovy and Hovy, 2012), temporal relation extraction (Ning et al., 2018), and dependency parsing (Flannery et al., 2012), and named entity recognition (Jie et al., 2019).

In particular, Jie et al. (2019) study a similar problem with a few key differences: since they remove entity surfaces randomly, the dataset is too easy; and they do not use constraints on their output. We compare against their results in our experiments.

Our proposed method is most closely aligned with the Constraint Driven Learning (CoDL) framework (Chang et al., 2007), in which an iterative algorithm reminiscent of self-training is guided by constraints that are applied at each iteration.

3.2.3. Constrained Binary Learning

Our method assigns instance weights to all negative elements, so that false negatives have low weights, and all other instances have high weights. We calculate weights according to the confidence predictions of a classifier trained iteratively over the partially annotated data. We refer to our method as Constrained Binary Learning (CBL).

We will first describe the motivation for this approach before moving on to the mechanics. Recall that we start with partially annotated data (which we call set T) in which some, but not all, positives are annotated (set P), and no negative is labeled. By default, we assume that any instance not labeled as positive is labeled as negative as opposed to unlabeled. This data (set N) is noisy in the sense that many true positives are labeled as negative (these are *false negatives*). Clearly, training on T as-is will result in a noisy classifier.

Two possible approaches are: **1)** find the false negatives and label them correctly, or **2)** find the false negatives and remove them. The former method affords more training data, but runs the risk of adding noise, which could be worse than the original partial annotations. The latter is more forgiving because of an asymmetry in the penalties: it is important to remove all false negatives in N, but inadvertently removing true negatives from N is typically not a problem, especially in NER, where negative examples dominate. Further, a

```
Require:
    $P$ : positive tokens
    $N$ : noisy negative tokens
    $C$ : constraints

1: $T = N \cup P$
2: $V \leftarrow$ Initialize $T$ with weights (Optional)
3: while stopping condition not met do
4:     $\lambda \leftarrow \text{train}(T, V)$
5:     $\hat{T} \leftarrow \text{predict}(\lambda, T)$
6:     $T, V \leftarrow \text{inference}(\hat{T}, C)$
7: end while
8: return $\lambda$
```

Figure 10: Constrained Binary Learning (CBL) algorithm. The core of the algorithm is in the while loop, which iterates over training on T, predicting on T and correcting those predictions.

binary model is sufficient in this case, as we need only detect entities, not type them.

We choose the latter method, but instead of removing false negatives, we adopt an instance-weighting approach, in which each instance is assigned a weight $v_i \geq 0$ according to confidence in the labeling of that instance. A weight of 0 means that the loss this instance incurs during training will not update the model.

With this in mind, CBL takes two phases: first, it learns a binary classifier λ using a constrained iterative process modeled after the CODL framework (Chang et al., 2007), and depicted in Figure 10. The core of the algorithm is the train-predict-infer loop. The training process (line 4) is weighted, using weights V. At the start, these can be all 1 (Raw), or can be initialized with prior knowledge. The learned model is then used to predict on all of T (line 5). In the inference step (line 6), we take the predictions from the prior round and the constraints C and produce a new labeling on T, and a new set of weights V. The details of this inference step are presented later in this section. Although our ultimate strategy is simply to assign weights (not change labels), in this inner loop, we update the labels on N according to classifier predictions.

In the second phase of CBL, we use the λ trained in the previous phase to assign weights to instances as follows:

$$v_i = \begin{cases} 1.0 & \text{if } x_i \in P \\ P_\lambda(y_i = 0 \mid x_i) & \text{if } x_i \in N \end{cases} \tag{3.1}$$

Where $P_\lambda(y_i = 0 \mid x_i)$ is understood as the classifier's confidence that instance x_i takes the negative label. In practice it is sufficient to use any confidence score from the classifier, not necessarily a probability.

Ultimately, we send the original multiclass partially annotated dataset along with final weights V to a standard weighted NER classifier to learn a model. No weights are needed at test time.

NER with CBL

So far, we have given a high-level view of the algorithm. In this section, we will give more low-level details, especially as they relate to the specific problem of NER. One of the most significant contributions of this work is the inference step (line 6), which we address using a constrained Integer Linear Program (ILP) and describe in this section. However, the constraints are based on a value we call the *entity ratio*. First, we describe the entity ratio, then we describe the constraints and stopping condition of the algorithm.

Entity ratio and Balancing We have observed that NER datasets tend to hold a relatively stable ratio of entity tokens to total tokens. We refer to this ratio as b, and define it with respect to some labeled dataset as:

$$b = \frac{|P|}{|P| + |N|} \tag{3.2}$$

where N is the set of negative examples. Previous work has shown that in fully-annotated datasets the entity ratio tends to be about 0.09 ± 0.05, depending on the dataset and genre (**?**). Intuitively, knowledge of the gold entity ratio can help us estimate when we have found all the false negatives.

In our main experiments, we assume that the entity ratio with respect to the gold labeling is known for each training dataset. A similar assumption was made in Elkan and Noto (2008) when determining the c value, and in Grave (2014) in the constraint determining the percentage of OTHER examples. However, we also show in Section 3.2.5 that knowledge of this ratio is not strictly necessary, and a flat value across all datasets produces similar performance.

With a weighted training set, it is also useful to define the weighted entity ratio.

$$b = \frac{|P|}{|P| + \sum_{i \in N} v_i} \tag{3.3}$$

When training an NER model on weighted data, one can change the weighted entity ratio to achieve different effects. To make balanced predictions on test, the entity ratio in the training data should roughly match that of the test data (Chawla, 2005). To bias a model towards predicting positives or predicting negatives, the weighted entity ratio can be set higher or lower respectively. This effect is pronounced when using linear methods for NER, but not as clear in neural methods.

To change the entity ratio, we scale the weights in N by a scaling constant γ. Targeting a particular b^*, we may write:

$$b^* = \frac{|P|}{|P| + \gamma \sum_{i \in N} v_i} \tag{3.4}$$

We can solve for γ:

$$\gamma = \frac{(1 - b^*)|P|}{b^* \sum_{i \in N} v_i} \tag{3.5}$$

To obtain weights, v_i^*, that attain the desired entity ratio, b^*, we scale all weights in N by γ.

$$v_i^* = \gamma v_i \tag{3.6}$$

In the train-predict-infer loop, we balance the weights to a value near the gold ratio before training.

Constraints and Stopping Condition We encode our constraints with an Integer Linear Program (ILP), shown in Figure 11. Intuitively, the job of the inference step is to take predictions ($\hat{T}$) and use knowledge of the task to 'fix' them.

In the objective function (Eqn. 3.8), token i is represented by two indicator variables y_{0i} and y_{1i}, representing negative and positive labels, respectively. Associated prediction scores C_0 and C_1 are from the classifier λ in the last round of predictions. The first constraint (Eqn. 3.9) encodes the fact that an instance cannot be both an entity and a non-entity.

The second constraint (Eqn. 3.10) enforces the ratio of positive to total tokens in the corpus to match a required entity ratio. $|T|$ is the total number of tokens in the corpus. b is the required entity ratio, which increases at each iteration. δ allows some flexibility, but is small.

Constraint 3.11 encodes that instances in P should be labeled positive since they were manually labeled and are by definition trustworthy. We set $\xi \geq 0.99$.

This framework is flexible in that more complex language- or task-specific constraints could be added. For simplicity, and because of the number of languages in our experiments, we use only a few constraints.

After the ILP has selected predictions, we assign weights to each instance in preparation for training the next round. The decision process for an instance is:

$$v_i = \begin{cases} 1.0 & \text{If ILP labeled } x_i \text{ positive} \\ P_\lambda(y_i = 0 \mid x_i) & \text{Otherwise} \end{cases} \tag{3.7}$$

This is similar to Equation (3.1), except that the set of tokens that the ILP labeled as

$$\begin{aligned}
\max_{\mathbf{y}} \quad & \sum_{i}^{|T|} C_{0i}y_{0i} + C_{1i}y_{1i} && (3.8)\\
\text{s.t.} \quad & \forall i,\ y_{0i} + y_{1i} = 1 && (3.9)\\
& b - \delta \leq \sum_i y_{1i}/|T| \leq b + \delta && (3.10)\\
& \forall i,\ x_i \in P,\ \sum_i y_{1i} \geq \xi|P|, && (3.11)
\end{aligned}$$

Figure 11: ILP for the inference step

positive is larger than P. With new labels and weights, we start the next iteration.

The stopping condition for the algorithm is related to the entity ratio. One important constraint (Eqn. 3.10) governs how many positives are labeled at each round. This number starts at $|P|$ and is increased by a small value[4] at each iteration, thereby improving recall. Positive instances are chosen in two ways. First, all instances in P are constrained to be labeled positive (Eqn. 3.11). Second, the objective function ensures that high-confidence positives will be chosen. The stopping condition is met when the number of required positive instances (computed using gold unweighted entity ratio) equals the number of predicted positive instances.

3.2.4. Experiments

We measure the performance of our method on 8 different languages using artificially perturbed labels to simulate the partial annotation setting.[5]

[4]The size of this value is related to how much we trust the ranking induced by prediction confidences. If we believed the ranking was perfect, we could take as many positives as we wanted and be finished in one round.

[5]We will release code and random seeds upon publication.

Data

We experiment on 8 languages. Four languages – English, German, Spanish, Dutch – come from the CoNLL 2002/2003 shared tasks (Sang, 2002; Sang and Meulder, 2003). These are taken from newswire text, and have labelset of Person, Organization, Location, Miscellaneous.

The remaining four languages come from the LORELEI project (Strassel and Tracey, 2016). These languages are: Amharic (amh: LDC2016E87), Arabic (ara: LDC2016E89), Hindi (hin: LDC2017E62), and Somali (som: LDC2016E91). These come from a variety of sources including discussion forums, newswire, and social media. The labelset is Person, Organization, Location, Geo-political entity. We define train/development/test splits, taking care to keep a similar distribution of genres in each split. Data statistics for all languages are shown in Table 3.

Artificial Perturbation

We create partial annotations by perturbing gold annotated data in two ways: lowering recall (to simulate missing entities), and lowering precision (to simulate noisy annotations).

To lower recall, we replace gold named entity tags with O tags (for non-name). We do this by grouping named entity surface forms, and replacing tags on all occurrences of a randomly selected surface form until the desired amount remains. For example, if the token 'Bangor' is chosen to be untagged, then every occurrence of 'Bangor' will be untagged. We chose this slightly complicated method because the simplest idea (remove mentions randomly) leaves an artificially large diversity of surface forms, which makes the problem of discovering noisy entities easier.

To lower precision, we tag a random span (of a random start position, and a random length between 1 and 3) with a random named entity tag. We continue this process until we reach the desired precision. When both precision and recall are to be perturbed, the recall adjustment is made first, and then the number of random spans to be added is calculated by the entities that are left. We record random seeds for all experiments so the data can be reproduced.

NER Models

In principle, CBL can use any NER method that can be trained with instance weights. We experiment with both non-neural and neural models.

Non-neural Model For our non-neural system, we use a version of Cogcomp NER (Ratinov and Roth, 2009; Khashabi et al., 2018b) modified to use Weighted Averaged Perceptron. This operates on a weighted training set $D_w = \{(x_i, y_i, v_i)\}_{i=1}^{N}$, where N is the number of training examples, and $v_i \geq 0$ is the weight on the ith training example. In this non-neural system, a training example is a word with context encoded in the features. We change only the update rule, where the learning rate α is multiplied by the weight:

$$\mathbf{w} = \mathbf{w} + \alpha v_i y_i (\mathbf{w}^T x_i) \tag{3.12}$$

We use a standard set of features, as documented in Ratinov and Roth (2009). In order to keep the language-specific resources to a minimum, we did not use any gazetteers for any language.[6] One of the most important features is Brown clusters, trained for 100, 500, and 1000 clusters for the CoNLL languages, and 2000 clusters for the remaining languages. We trained these clusters on Wikipedia text for the four CoNLL languages, and on the same monolingual text used to train the word vectors (described in Section 3.2.4).

Neural Model A common neural model for NER is the BiLSTM-CRF model (Ma and Hovy, 2016). However, because the Conditional Random Field (CRF) layer calculates loss at the sentence level, we need a different method to incorporate token weights. We use a variant of the CRF that allows partial annotations by marginalizing over all possible

[6]Separate experiments show that omitting gazetteers impacts performance only slightly.

sequences (Tsuboi et al., 2008).

When using a standard BiLSTM-CRF model, the loss of a dataset is calculated as:

$$\mathcal{L} = -\sum_{s \in D} \log P_\theta(\mathbf{y}^{(s)} | \mathbf{x}^{(s)}) \tag{3.13}$$

Where $P_\theta(\mathbf{y}^{(s)}|\mathbf{x}^{(s)})$ is calculated by the CRF over outputs from the BiLSTM. In the marginal CRF framework, it is assumed that $\mathbf{y}^{(s)}$ is necessarily partial, denoted as $\mathbf{y}_p^{(s)}$. To incorporate partial annotations, the loss is calculated by marginalizing over all possible sequences consistent with the partial annotations, denoted as $C(\mathbf{y}_p^s)$.

$$\mathcal{L} = -\sum_{s \in D} \log \sum_{\mathbf{y} \in C(\mathbf{y}_p^{(s)})} P_\theta(\mathbf{y} | \mathbf{x}^{(s)}) \tag{3.14}$$

However, this formulation assumes that all possible sequences are equally likely. To address this, Jie et al. (2019) introduced a way to weigh sequences.

$$\mathcal{L} = -\sum_{s \in D} \log \sum_{\mathbf{y} \in C(\mathbf{y}_p^{(s)})} q(\mathbf{y} | \mathbf{x}^{(s)}) P_\theta(\mathbf{y} | \mathbf{x}^{(s)}) \tag{3.15}$$

It's easy to see that this formulation is a generalization of the standard CRF if $q(.) = 1$ for the gold sequence $\mathbf{y}$, and 0 for all others.

The product inside the summation depends on tag transition probabilities and tag emission probabilities, as well as token-level "weights" over the tagset. These weights can be seen as defining a soft labeling for each token, corresponding to confidence in each label.

We incorporate our instance weights in this model with the following intuitions. Recall that if an instance weight $v_i = 0$, this indicates low confidence in the label on token x_i, and therefore the labeling should not update the model at training time. Conversely, if $v_i = 1$,

	Train			Test		
Lang.	b (%)	Tag	Tok	b (%)	Tag	Tok
English	16.6	23K	203K	17.3	5K	46K
Spanish	12.3	18K	264K	11.9	3K	51K
German	8.0	11K	206K	9.9	3K	51K
Dutch	9.5	13K	202K	8.3	4K	68K
Amharic	11.2	3K	52K	11.3	1K	18K
Arabic	12.6	4K	60K	10.2	931	16K
Hindi	7.38	4K	74K	7.53	1K	25K
Somali	11.2	4K	57K	11.9	1K	16K

Table 3: Data statistics for all languages, showing number of tags and tokens in Train and Test. The tag counts represent individual spans, not tokens. That is, "[Barack Obama]$_{\text{PER}}$" counts as one tag, not two. The b column shows the entity ratio as a percentage.

then this label is to be trusted entirely.

If $v_i = 0$, we set the soft labeling weights over x_i to be uniform, which is as good as no information. Since v_i is defined as confidence in the O label, the soft labeling weight for O increases proportionally to v_i. Any remaining probability mass is distributed evenly among the other labels.

We use pretrained GloVe (Pennington et al., 2014) word vectors for English, and the same pretrained vectors used in Lample et al. (2016) for Dutch, German, and Spanish. The other languages are distributed with monolingual text (Strassel and Tracey, 2016), which we used to train our own skip-n-gram vectors.

Baselines

We compare against several baselines, including two from prior work.

Raw annotations The simplest baseline is to do nothing to the partially annotated data and train on it as is.

Instance Weights Although CBL works with no initialization (that is, all tokens with weight 1), we found that a good weighting scheme can boost performance for certain models.

Method \ Lang.	Tool	**eng**	**deu**	**esp**	**ned**	**amh**	**ara**	**hin**	**som**	avg
Gold	Cogcomp	89.1	72.5	82.5	82.6	67.2	53.4	74.4	80.3	75.3
	BiLSTM-CRF	90.3	77.3	85.2	81.1	69.2	52.8	73.8	82.3	76.5
Oracle Weighting	Cogcomp	83.7	65.7	76.2	76.4	54.3	42.0	56.3	68.5	65.4
	BiLSTM-CRF	87.8	70.2	78.5	70.4	60.4	43.4	57.6	73.2	67.7
Noise Adaptation (Hedderich, 2018)		61.5	46.1	57.3	41.5	–	–	–	–	–
Self-training (Jie et al., 2019)		82.3	65.2	76.3	65.5	52.1	40.1	55.1	65.3	62.7
Raw Annotations	Cogcomp	54.8	36.9	49.5	47.9	31.0	32.6	30.9	44.0	40.9
	BiLSTM-CRF	73.3	57.7	61.9	58.3	42.2	36.8	47.5	54.9	54.1
CBL Raw	CogComp	74.7	63.0	68.7	67.0	45.0	37.8	50.6	67.9	59.3
	BiLSTM-CRF	**84.6**	**67.9**	**79.6**	70.0	**52.9**	42.1	55.2	**70.4**	**65.3**
Comb. Weighting	Cogcomp	75.2	56.6	70.8	70.8	46.5	**44.1**	57.5	60.2	60.2
	BiLSTM-CRF	73.5	60.3	64.9	61.9	48.0	38.0	49.0	56.6	56.5
CBL Combined	Cogcomp	77.3	61.8	74.0	**72.4**	49.2	43.7	**58.2**	67.6	63.0
	BiLSTM-CRF	81.1	64.9	74.9	63.4	52.2	39.8	52.0	67.0	61.9

Table 4: F1 scores on English, German, Spanish, Dutch, Amharic, Arabic, Hindi, and Somali. Each section shows performance of both Cogcomp (non-neural) and BiLSTM (neural) systems. *Gold* is using all available gold training data to train. *Oracle Weighting* uses full entity knowledge to set weights on N. The next section shows prior work, followed by our methods. The column to the farthest right shows the average score over all languages. Bold values are the highest per column. On average, our best results are found in the uninitialized (*Raw*) CBL from BiLSTM-CRF.

We design weighting schemes that give instances in N weights corresponding to an estimate of the label confidence.[7] For example, non-name tokens such as *respectfully* should have weight 1, but possible names, such as *Russell*, should have a low weight, or 0. We propose two weighting schemes: frequency-based and window-based.

For the frequency-based weighting scheme, we observed that names have relatively low frequency (for example, *Kennebunkport*, *Dushanbe*) and common words are rarely names (for example *the*, *and*, *so*). We weigh each instance in N according to its frequency.

$$v_i^{\text{freq}} = freq(x_i) \tag{3.16}$$

where $freq(x_i)$ is the frequency of the i^{th} token in N divided by the count of the most frequent token. In our experiments, we computed frequencies over $P + N$, but these could

[7]All elements of P always have weight 1

be estimated on any sufficiently large corpus. We found that the neural model performed poorly when the weights followed a Zipfian distribution (e.g. most weights very small), so for those experiments, we took the log of the token count before normalizing.

For the window-based weighting scheme, noting that names rarely appear immediately adjacent to each other in English text, we set weights for tokens within a window of size 1 of a name (identified in P) to be 1.0, and for tokens farther away to be 0.

$$v_i^{\text{window}} = \begin{cases} 1.0 & \text{if } d_i \leq 1 \\ 0.0 & \text{otherwise} \end{cases} \tag{3.17}$$

where d_i is the distance of the i^{th} token to the nearest named entity in P.

Finally, we combine the two weighting schemes as:

$$v_i^{\text{combined}} = \begin{cases} 1.0 & \text{if } d_i \leq 1 \\ v_i^{\text{freq}} & \text{otherwise} \end{cases} \tag{3.18}$$

Self-training with Marginal CRF Jie et al. (2019) propose a model based on marginal CRF (Tsuboi et al., 2008) (described in Section 3.2.4). They follow a self-training framework, using the trained model from the previous round to update gold labeling distributions at each round.

Neural Network with Noise Adaptation Following (Hedderich and Klakow, 2018), we used a neural network with a noise adaptation layer.[8] This extra layer attempts to correct noisy examples given a probabilistic confusion matrix of label noise. Since this method needs a small amount of labeled data, we selected 500 random tokens to be the gold training set, in addition to the partial annotations.

[8]The code was kindly provided by the authors.

As with our BiLSTM experiments, we use pretrained GloVe word vectors for English, and the same pretrained vectors used in (Lample et al., 2016) for Dutch, German, and Spanish. We omit results from the remaining languages because the scores were substantially worse even than training on raw annotations.

Experimental Setup and Results

We show results from our experiments in Table 4. In all experiments, the training data is perturbed at 90% precision and 50% recall. These parameters are similar to the scores obtained by human annotators in a foreign language (see Section 3.2.6). We evaluate each experiment with both non-neural and neural methods.

First, to get an idea of the difficulty of NER in each language, we report scores from models trained on gold data without perturbation (*Gold*). Then we report results from an Oracle Weighting scheme (*Oracle Weighting*) that takes partially annotated data and assigns weights with knowledge of the true labels. Specifically, mislabeled entities in set N are given weight 0, and all other tokens are given weight 1.0. This scheme is free from labeling noise, but should still get lower scores than Gold because of the smaller number of entities. Since our method estimates these weights, we do not expect CBL to outperform the Oracle method. Next, we show results from all baselines. The bottom two sections are our results, first with no initialization (*Raw*), and CBL over that, then with *Combined Weighting* initialization, and CBL over that.

Analysis

Regardless of initialization or model, CBL improves over the baselines. Our best model, *CBL-Raw BiLSTM-CRF*, improves over the *Raw Annotations BiLSTM-CRF* baseline by 11.2 points F1, and the *Self-training* prior work by 2.6 points F1, showing that it is an effective way to address the problem of partial annotation. Further, the best CBL version for each model is within 3 points of the corresponding *Oracle* ceiling, suggesting that this weighting framework is nearly saturated.

The *Combined* weighting scheme is surprisingly effective for the non-neural model, which suggests that the intuition about frequency as distinction between names and non-names holds true. It gives modest improvement in the neural model. The *Self-training* method is effective, but is outperformed by our best CBL method, showing the importance of using constraints. The *Noise Adaptation* method outperforms the *Raw annotations Cogcomp* baseline in most cases, but does not reach the performance of the *Self-training* method, despite using some fully labeled data.

It is instructive to compare the neural and non-neural versions of each setup. The neural method is better overall, but is less able to learn from the knowledge-based initialization weights. In the non-neural method, the difference between *Raw* and *Combined* is nearly 20 points, but the difference in the neural model is less than 3 points. *Combined* versions of the non-neural method outperform the neural method on 3 languages: Dutch, Arabic, and Hindi. Further, in the neural method, *CBL-Raw* is always better than *CBL-Combined*. This may be due to the way that weights are used in each model. In the non-neural model, a low enough weight completely cancels the token, whereas in the neural model it is still used in training. Since the neural model performs well in the *Oracle* setting, we know that it can learn from hard weights, but it may have trouble with the subtle differences encoded in frequencies. We leave it to future work to discover improved ways of incorporating instance weights in a BiLSTM-CRF.

In seeking to understand the details of the other results, we need to consider the precision/recall tradeoff. First, all scores in the *Gold* row had higher precision than recall. Then, training on raw partially annotated data biases a classifier strongly towards predicting few entities. All results from the *Raw annotations* row have precision more than double the recall (e.g. Dutch Precision, Recall, F1 were: 91.5, 32.4, 47.9). In this context, the problem this paper explores is how to improve the recall of these datasets without harming the precision.

	Avg F1		
Method \ b	10%	15%	Gold
Oracle Weighting	65.8	**65.9**	65.4
Raw annotations	40.9	40.9	40.9
Combined Weighting	59.9	**60.2**	**60.2**
CBL-Combined	62.4	62.3	**63.0**

Table 5: Experimenting with different entity ratios. Scores reported are average F1 across all languages. *Gold b* value refers to using the gold annotated data to calculate the optimal entity ratio. This table shows that exact knowledge of the entity ratio is not required for CBL to succeed.

3.2.5. Varying the Entity Ratio

Recall that the entity ratio is used for balancing and for the stopping criteria in CBL. In all our experiments so far, we have used the gold entity ratio for each language, as shown in Table 3. However, exact knowledge of entity ratio is unlikely in the absence of gold data. Thus, we experimented with selecting a default b value, and using it across all languages, with the Cogcomp model. We chose values of 10% and 15%, and report F1 averaged across all languages in Table 5.

While the gold b value is the best for *CBL-Combined*, the flat 15% ratio is best for all other methods, showing that exact knowledge of the entity ratio is not necessary.

3.2.6. Bengali Case Study

So far our experiments have shown effectiveness on artificially perturbed labels, but one might argue that these systematic perturbations don't accurately simulate real-world noise. In this section, we show how our methods work in a real-world scenario, using Bengali data partially labeled by non-speakers.

Num tokens	49K
Num sentences	2435
Num name tokens	2326
Entity ratio	4.66%
Num unique name tokens	664
Annotator 1 Prec/Rec/F1	84/34/48
Annotator 2 Prec/Rec/F1	79/28/42
Combined Prec/Rec/F1	83/32/47

Table 6: Bengali Data Statistics. The P/R/F1 scores are computed for the non-speaker annotator with respect to the gold training data.

Non-speaker Annotations

In order to compare with prior work, we used the train/test split from Zhang et al. (2016). We removed all gold labels from the train split, romanized it[9], and presented it to two non-Bengali speaking annotators. The instructions were to move quickly and annotate names only when there is high confidence (e.g. when you can also identify the English version of the name). They spent about 5 total hours annotating, without using Google Translate. This sort of non-speaker annotation is possible because the text contains many 'easy' entities – foreign names – which are noticeably distinct from native Bengali words. For example, consider the following:

- **Romanized Bengali**: ebisi'ra giliyyaana phinnddale aaja pyaalestaaina adhiinastha gaajaa theke aaja raate ekhabara jaaniyyechhena .
- **Translation**[10]: ABC's Gillian Fondley has reported today from Gaza under Palestine today.

The entities are Gillian Findlay, ABC, Palestine, and Gaza. While a fast-moving annotator may not catch most of these, 'pyaalestaaina' could be considered an 'easy' entity, because of its visual and aural similarity to 'Palestine.' A clever annotator may also infer that if Palestine is mentioned, then Gaza may be present.

[9]This step is vitally important. We used `www.isi.edu/~ulf/uroman.html`

[10]From `translate.google.com`

	Test		
Scheme	P	R	F1
(Zhang et al., 2016)	-	-	34.8
(Tsai et al., 2016)	-	-	43.3
(Pan et al., 2017)	-	-	44.0
(Mayhew et al., 2017b)	-	-	46.2
BiLSTM-CRF			
Train on Gold	71.6	70.2	70.9
Raw annotations	73.0	23.8	35.9
Combined Weighting	65.9	34.2	45.0
CBL-Raw	57.8	**47.3**	**52.0**
CBL-Combined	58.3	44.2	50.2

Table 7: Bengali manual annotation results. Our methods improve on state of the art scores by over 5 points F1 given a relatively small amount of noisy and incomplete annotations from non-speakers.

Annotators are moving fast and being intentionally non-thorough, so the recall will be low. Since they do not speak Bengali, there are likely to be some mistakes, so the precision may drop slightly also. This is exactly the noisy partial annotation scenario addressed in this paper. The statistics of this data can be seen in Table 6, including annotation scores computed with respect to the gold training data for each annotator, as well as the combined score.

We show results in Table 7, using the BiLSTM-CRF model. We compare against other low-resource approaches published on this dataset, including two based on Wikipedia (Tsai et al., 2016; Pan et al., 2017), another based on lexicon translation from a high-resource language (Mayhew et al., 2017b). These prior methods operate under somewhat different paradigms than this work, but have the same goal: maximizing performance in the absence of gold training data.

Raw annotations is defined as before, and gives similar high-precision low-recall results. The *Combined Weighting* scheme improves over Raw annotations by 10 points, achieving a score comparable to the prior state of the art. Beyond that, *CBL-Raw* outperforms the prior best by nearly 6 points F1, although *CBL-Combined* again underwhelms.

To the best of our knowledge, this is the first result showing a method for non-speaker annotations to produce high-quality NER scores. The simplicity of this method and the small time investment for these results gives us confidence that this method can be effective for many low-resource languages.

3.2.7. Conclusions

We explore an understudied data scenario, and introduce a new iterative algorithm to solve it. Our contributions include suggestions for weighting schemes and and an iterative constrained algorithm, all of which perform well in experimental trials, showing that we successfully address the problem of partial annotation. We have shown that our methods are robust on artificially perturbed labels, and can further give promising results in a truly low-resource situation.

3.3. The Internal Structure of Name Tokens: A Multilingual Study

This work published as (Yu et al., 2018).

Character-level patterns have been widely used as features in English Named Entity Recognition (NER) systems. However, to date there has been no direct investigation of the inherent differences between name and non-name tokens in text, nor whether this property holds across multiple languages.

This paper analyzes the capabilities of corpus-agnostic Character-level Language Models (CLMs) in the binary task of distinguishing name tokens from non-name tokens. We demonstrate that CLMs provide a simple and powerful model for capturing these differences, identifying named entity tokens in a diverse set of languages at close to the performance of full NER systems. Moreover, by adding very simple CLM-based features we can significantly improve the performance of an off-the-shelf NER system for multiple languages.

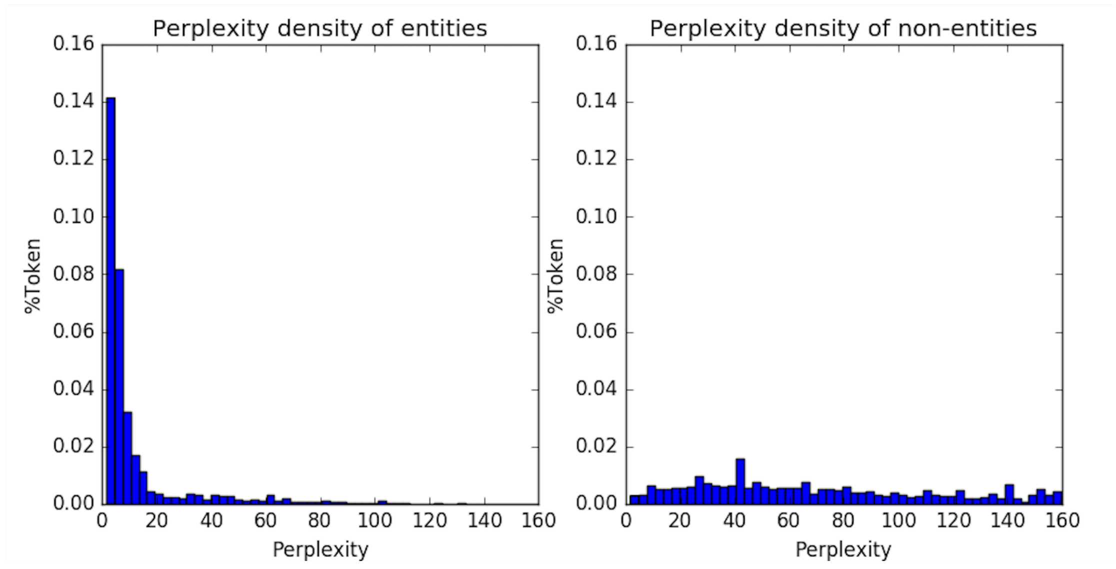


Figure 12: Perplexity histogram of entity (left) and non-entity tokens (right) in CoNLL Train calculated by entity CLM for both sides. The graphs show the percentage of tokens (Y axis) with different levels of CLM perplexities. The entity CLM gives a low average perplexity and small variance to entity tokens (left), while giving non-entity tokens much higher perplexity and higher variance (right).

3.3.1. Introduction

In English, there is strong empirical evidence that the character sequences that make up proper nouns tend to be distinctive.

Even divorced of context, a human reader can predict that “hoekstenberger” is an entity, but “abstractually”[11] is not. Some NER research explores the use of character-level features including capitalization, prefixes and suffixes (Cucerzan and Yarowsky, 1999; Ratinov and Roth, 2009), and character-level language models (CLMs) (Klein et al., 2003) to improve the performance of NER, but to date there has been no systematic study isolating the utility of CLMs in capturing distinctions between name and non-name tokens in English or across other languages.

We conduct an experimental assessment of the discriminative power of CLMs for a range of languages: English, Amharic, Arabic, Bengali, Farsi, Hindi, Somali, and Tagalog. These

[11]Not a real name or a real word.

Model	**eng**	**amh**	**ara**	**ben**	**fas**	**hin**	**som**	**tgl**	avg
Exact Match	43.4	54.4	29.3	47.7	30.5	30.9	46.0	23.7	37.5
Capitalization	79.5	-	-	-	-	-	69.5	77.6	-
SRILM	**92.8**	**69.9**	**54.7**	**79.4**	**60.8**	**63.8**	**84.1**	**80.5**	**70.5**
Skip-gram	76.0	53.0	29.7	41.4	30.8	29.0	51.1	61.5	42.4
CBOW	73.7	50.0	28.1	40.6	32.6	26.5	56.4	62.5	42.4
Log-Bilinear	82.8	64.5	46.1	70.8	50.4	54.8	78.1	74.9	62.8
CogCompNER (ceiling)	96.5	73.8	64.9	80.6	64.1	75.9	89.4	88.6	76.8
(Lample et al., 2016) (ceiling)	96.4	84.4	69.8	87.6	76.4	86.3	90.9	91.2	83.8

Table 8: Token level identification F1 scores. Averages are computed over all languages other than English. Two baselines are also compared here: *Capitalization* tags a token in test as entity if it is capitalized, and *Exact Match* keeps track of entities seen in training and tags tokens in Test that exactly match some entity in Train. The bottom section shows state of the art models which use complex features for names and also their contexts. Languages in order are: English, Amharic, Arabic, Bengali, Farsi, Hindi, Somali, and Tagalog.

languages use a variety of scripts and orthographic conventions (for example, only three use capitalization), come from different language families, and vary in their morphological complexity. We demonstrate the effectiveness of CLMs in distinguishing name tokens from non-name tokens, as illustrated by Figure 12, which shows perplexity histograms from a CLM trained on entity tokens. Our models use only individual tokens, but perform extremely well in spite of taking no account of word context.

We then assess the utility of directly adding simple features based on this CLM implementation to an existing NER system, and show that they have a significant positive impact on performance across many of the languages we tried. By adding very simple CLM-based features to the system, our scores approach those of a state of the art NER system (Lample et al., 2016) across multiple languages, demonstrating both the unique importance and the broad utility of this approach.

3.3.2. Methods

Character Language Models

We propose a very simple model in which we train an entity CLM on a list of entity tokens, and a non-entity CLM on a list of non-entity tokens. Both lists are unordered, with all entries treated independently. Each entity (and non-entity) token is split into characters, and treated as a “sentence” where the characters are the “words”, and we learn a score measuring how likely it is that a sequence of characters forms an entity. For example, “Obama” is an entity token, and is split into “O b a m a”. At test time, we also split each word into characters and determine perplexity using the entity and non-entity CLMs. We assign the label corresponding to the lower perplexity CLM.

We experiment with four different kinds of language model: N-gram model, Skip-gram model, Continuous Bag-of-Words model (CBOW), and Log-Bilinear model (LB), and demonstrate that the N-gram model is best suited for this task.

Following (Peng and Roth, 2016), we implement N-gram using SRILM (Stolcke, 2002) with order 6 and Witten-Bell discounting.[12] For Skip-Gram and CBOW CLMs, we use the Gensim implementation (Řehůřek and Sojka, 2010) for training and inference, and we build the LB CLM using the OxLM toolkit (Baltescu et al., 2014).

Data

To determine whether name identifiability applies to languages other than English, we conduct experiments on a range of languages for which we had previously gathered resources (such as Brown clusters): English, Amharic, Arabic, Bengali, Farsi, Hindi, Somali, and Tagalog.

For English, we use the ubiquitous CoNLL 2003 English dataset (Tjong Kim Sang and

[12]We experimented with different orders on development data, but found little difference between them.

De Meulder, 2003), which is a newswire dataset annotated with Person (PER), Organization (ORG), Location (LOC) and Miscellaneous (MISC). To collect the list of entities and non-entities as the training data for the Entity and Non-Entity CLMs, we sample a large number of PER/ORG/LOC and non-entities from Wikipedia, using types derived from their corresponding FreeBase entities (Ling and Weld, 2012b).

For all other languages, we use a subset of the corpora from the LORELEI project annotated for the NER task (Strassel and Tracey, 2016).[13] We build our entity list using the tokens labeled as entities in the training data, and our non-entity list from the remaining tokens. These two lists are then used to train two CLMs, as described above.

Our datasets vary in size of entity and non-entity tokens. The smallest, Farsi, has 4.5K entity and 50K non-entity tokens; the largest, English, has 29K entity and 170K non-entity tokens.

3.3.3. Character Language Models for Named Entity Identification

In this section, we first show the power of CLMs for distinguishing between entity and non-entity tokens in English, and then that this power is robust across a variety of languages.

We refer to this task as Named Entity Identification (NEI), because we are concerned only with finding an entity span, not its label. We differentiate it from Named Entity Recognition (NER), in which both span and label are required. To avoid complicating this straightforward approach by requiring a separate mention detection step, we evaluate at the token level. We also apply one heuristic: if a word has length 1, we automatically predict 'O'.

Figure 1 shows that for the majority of entity tokens, the entity CLM computes a relatively low perplexity compared to non-entity tokens. Though there also exist some non-entities with low entity CLM perplexity, we can still reliably identify a large proportion of non-entity words by setting a threshold value for entity CLM perplexity. If a token perplexity

[13]We create the first train/test split of this data, and will publish it once anonymity is no longer required.

Model		eng	amh	ara	ben	fas	hin	som	tgl	avg
(Lample et al., 2016)	Full	90.94	**73.2**	57.2	**77.7**	**61.2**	**77.7**	81.3	**83.2**	**73.1**
	Unseen	86.11	51.9	30.2	57.9	41.4	62.2	66.5	72.8	54.7
CogCompNER	Full	90.88	67.5	54.8	74.5	57.8	73.5	82.0	80.9	70.1
	Unseen	84.40	42.7	25.0	51.9	31.5	53.9	67.2	68.3	48.6
CogCompNER+LM	Full	**91.21**	71.3	**59.1**	75.5	59.0	74.2	**82.1**	78.5	71.4
	Unseen	85.20	48.4	32.0	54.0	31.2	55.4	68.0	65.2	50.6

Table 9: NER results on 8 languages show that even a simplistic addition of CLM features to a standard NER model boosts performance. CogCompNER is run with standard features, including Brown clusters; (Lample et al., 2016) is run with default parameters and pre-trained embeddings. *Unseen* refers to performance on named entities in Test that were not seen in the training data. *Full* is performance on all entities in Test. Averages are computed over all languages other than English.

lies above this threshold, we label it as a non-entity token. The threshold is tuned on development data.

Since we also build a CLM for non-entities, we can also compare the entity and non-entity perplexity scores for a token. For those tokens not excluded using the threshold as described above, we compare the perplexity scores of the two models and assign the label corresponding to the model yielding the lower score.

We compare SRILM against Skip-gram and CBOW, as implemented in Gensim, and the Log-Bilinear (LB) model. We trained both CBOW and Skip-gram with window size 3, and size 20. We tuned LB, and report results with embedding size 150, and learning rate 0.1. Despite tuning the neural models, the simple N-gram model outperforms them significantly, perhaps because of the relatively small amount of training data.[14]

We compare the CLM's Entity Identification against two state-of-the-art NER systems, anonymizing all the labels and evaluating on token level. As Table 8 shows, the result of N-gram CLM, which yields the highest performance, is remarkably close to the result of state-of-the-art NER systems (Lample et al., 2016), and can be trained in seconds.

[14]We also tried a simple RNN+GRU language model, but found that the results were also underwhelming.

3.3.4. Improving NER with CLM features

In this section we show that we can augment a standard NER system with simple features based on our entity/non-entity CLMs to improve performance in many languages. Based on their superior performance as reported in Section 3.3.3, we use the N-gram CLMs.

Features

We define three simple features that capture information provided by CLMs and which we expect to be useful for NER.

Entity Features We define one "isEntity" feature based on the perplexities of the entity and non-entity CLMs. We compare the perplexity calculated by entity CLM and non-entity CLM described in Section 3.3.3, and return a boolean value indicating whether the entity CLM score is lower.

Language Features We define two language-related features: "isArabic" and "isRussian". We observe that there are many names in English text that originate from other languages, resulting in very different orthography than native English names. We therefore build two language-based CLMs for Arabic and Russian. We collect a list of Arabic names and a list of Russian names by scraping name-related websites, and train an Arabic CLM and a Russian CLM. For each token, when the perplexity of either the Arabic or the Russian CLM is lower than the perplexity of the Non-Entity CLM, we return True, indicating that this entity is likely to be a name from Arabic/Russian. Otherwise, we return False.

Experiments

We use CogCompNER (Ratinov and Roth, 2009; Khashabi et al., 2018a) as our baseline NER system because it allows easy integration of new features, and evaluate on the same datasets as before. For English, we add all features described above. For other languages, due to the limited training data, we only use the "isEntity" features. We compare with the state-of-the-art character-level neural NER system of (Lample et al., 2016), which inherently encodes comparable information to CLMs, as a way to investigate how much of that system's performance can be attributed directly to name-internal structure.

The results in Table 9 show that for six of the eight languages we studied, the baseline NER can be significantly improved by adding simple CLM features; for English and Arabic, it performs better even than the neural NER model of (Lample et al., 2016). For Tagalog, however, adding CLM features actually impairs system performance.

In the same table, the rows marked “unseen” report systems' performance on named entities in Test that were not seen in the training data. This setting more directly assesses the robustness of a system to identify named entities in new data. By this measure, Farsi NER is not improved by name-only CLM features and Tagalog is impaired. Benefits for English, Hindi, and Somali are limited, but are quite significant for Amharic, Arabic, and Bengali.

3.3.5. Discussion

Our results demonstrate the power of CLMs for recognizing named entity tokens in a diverse range of languages, and that in many cases they can improve off-the-shelf NER system performance even when integrated in a simplistic way.

However, the results from Section 3.3.4 show that this is not true for all languages, especially when only considering unseen entities in Test: Tagalog and Farsi do not follow the trend for the other languages we assessed even though CLM performs well for Named Entity Identification.

While the end-to-end model developed by (Lample et al., 2016) clearly includes information comparable to that in the CLM, it requires a fully annotated NER corpus, takes significant time and computational resources to train, and is non-trivial to integrate into a new NER system. The CLM approach captures a very large fraction of the entity/non-entity distinction capacity of full NER systems, and can be rapidly trained using only entity and non-entity token lists – i.e., it is corpus-agnostic. For some languages it can be used directly to improve NER performance; for others (such as Tagalog), the strong NEI performance indicates that while it does not immediately boost performance, it can ultimately be used to improve NER there too.

3.3.6. Conclusions

We have shown, in a series of simple experiments, that in many languages names are identifiable by character patterns alone, and that character level patterns have strong potential for building better NER systems.

CHAPTER 4 : Cross-lingual NER

Having discussed non-traditional low-resource methods such as non-speaker annotations, partial annotations, and character language models, we now move to more standard cross-lingual techniques. We first discuss a Wikipedia-based method for cross-lingual NER, and finish the chapter with a work on "Cheap Translation" of high-resource annotations into low-resource languages using bilingual lexicons.

4.1. Cross-lingual NER via Wikification

This work published as (Tsai et al., 2016).

In this work, we take advantage of the rich multilingual resource that is Wikipedia. The key contribution is the development of a method to use cross-lingual wikification and entity linking (Tsai and Roth, 2016a; Ji et al., 2015, 2016; Moro et al., 2014) to generate language-independent features for NER, which can be used in a direct transfer setting.

Figure 13 shows an example of a German sentence. We use a cross-lingual wikifier to ground each word to the English Wikipedia (not all words have groundings). We can see that even though the disambiguation is not perfect, the FreeBase types still provide valuable information. That is, although "Albrecht Lehmann" is not an entry in Wikipedia, the wikifier still links "Albrecht" and "Lehmann" to people. Since words in any language are grounded to the English Wikipedia, the corresponding Wikipedia categories and Freebase types can be used as language-independent features.

The proposed model significantly outperforms comparable direct transfer methods on the Spanish, Dutch, and German CoNLL data. We also evaluate the model on five low-resource languages: Turkish, Tagalog, Yoruba, Bengali, and Tamil. Due to small sizes of Wikipedia, the overall performance is not as good as the CoNLL experiments. Nevertheless, the wikifier features still give significant improvements, and the proposed direct transfer model outperforms the state of the art, which assumes parallel text and some interaction with a

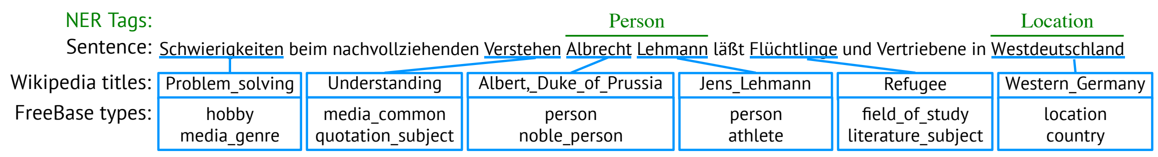


Figure 13: An example of a German sentence. We ground each word to the English Wikipedia using a cross-lingual wikifier. A word is not linked if it is a stop word or the wikifier returns NIL. We can see that the FreeBase types are strong signals to NER even with imperfect disambiguation.

native speaker of the target language. In addition, we show that the proposed language-independent features not only perform well on the direct transfer scenario, but also improve monolingual models, which are trained on the target language. Another advantage of the proposed direct transfer model is that we can train on documents from multiple languages together, and further improve the results.

4.1.1. Named Entity Recognition Model

We use the state of the art English NER model of Ratinov and Roth (2009) as the base model, with standard features (which we refer to as Base features), and the novel addition of wikifier features.

Cross-lingual Wikifier Features

As shown in Figure 13, disambiguating words to Wikipedia entries allows us to obtain useful information for NER from the corresponding FreeBase types and Wikipedia categories. A cross-lingual wikifier grounds words and phrases of non-English languages to the English Wikipedia, which provides language-independent features for transferring an NER model directly.

We use the system proposed in Tsai and Roth (2016a), which grounds input strings to the intersection of (the title spaces of) the English and the target language Wikipedias. The only requirement is a multilingual Wikipedia dump and it can be applied to all languages in Wikipedia.

	Latin Script							Non-Latin Script		
APPROACH	EN	NL	DE	ES	TR	TL	YO	BN	TA	AVG
Wiki size	5.1M	1.9M	1.9M	1.3M	269K	64K	31K	42K	85K	-
En. intersection	-	755K	964K	757K	169K	49K	30K	34K	51K	-
Gazetteer size	8.5M	579K	1M	943K	168K	54K	20K	29K	10K	-
Entities (train)	23.5K	18.8K	11.9K	13.3K	5.1K	4.6K	4.1K	8.8K	7.0K	-
Entities (test)	5.6K	3.6K	3.7K	3.9K	2.2K	3.4K	3.4K	3.5K	4.6K	-
Monolingual Experiments										
Wikifier only	71.57	57.02	49.74	60.13	52.84	51.02	29.35	47.78	38.05	50.83
Base Features	85.50	76.64	65.88	80.66	64.98	75.03	55.26	69.26	55.93	69.90
+Gazetteers	89.49	82.41	69.31	83.62	70.41	76.71	57.12	69.51	57.10	72.89
+Wikifier	**89.92**	**84.49**	**73.13**	**83.87**	**73.86**	**77.64**	**57.60**	**71.15**	**60.02**	**74.72**
Direct Transfer Experiments										
Wikifier only		40.44	39.83	43.82	41.79	42.11	27.91	**43.27**	**29.64**	38.01
Base Features		43.38	24.93	42.85	29.21	49.85	32.57	2.53	1.74	28.06
+Gazetteers		50.26	34.47	54.59	30.21	64.06	34.37	3.25	0.30	33.83
+Wikifier		**61.56**	**48.12**	**60.55**	**47.12**	**65.44**	**36.65**	18.18	5.65	**41.41**
Täckström baseline		48.4	23.5	45.6	-	-	-	-	-	-
Täckström bitext clusters		58.4	40.4	59.3	-	-	-	-	-	-
(Zhang et al., 2016)		-	-	-	43.6	51.3	36.0	34.8	26.0	38.3

Table 10: **Data sizes, monolingual experiments, and direct transfer experiments**. Wiki size is the number of articles in Wikipedia. For monolingual experiments, we train the proposed model on the training data of the target languages. 'Wikifier only' uses the previous tags features also. For direct transfer experiments, all models are trained on CoNLL English training set. The rows marked Täckström come from Täckström et al. (2012), and are the baseline and clustering result. The plus signs (+) signify cumulative addition. EN: English, NL: Dutch, DE: German, ES: Spanish, TR: Turkish, TL: Tagalog, YO: Yoruba, BN: Bengali, TA: Tamil.

After wikifying every n-gram [1], we set the types of each n-gram as the coarse- and fine-grained FreeBase types and Wikipedia categories from the top 2 title candidates returned by wikifier. For each word w_i, we use the types of w_i, w_{i+1}, and w_{i-1}, and the types of the n-grams which contain w_i as features. Moreover, we also include wikifier's ranking features from the top candidate as features. This could serve as a linker, which rejects the top prediction if it has a low confidence.

4.1.2. Experiments and Analysis

In this section, we conduct experiments to validate and analyze the proposed NER model. First, we show that adding wikifier features improves results on monolingual NER. Second, we show that wikifier features are strong signals in direct transfer of a trained NER model across languages. Finally, we explore the importance of Wikipedia size to the quality of wikifier features and study the use of multiple source languages.

To evaluate, we use several standard datasets from the CoNLL2002/2003 shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003), and some new datasets from the LORELEI project.

Monolingual Experiments

We begin by showing that wikifier features help when we train and test on the same language. The middle section of Table 12 shows these results.

In the 'Wikifier only' row, we use only wikifier features and previous tags features. This is intended to show the predictive power of wikifier features alone. Without using any lexical features, it gets good scores on the languages that have a large Wikipedia. These numbers represent the quality of the cross-lingual wikifier in that language, which in turn is correlated with the size of Wikipedia and size of the intersection with English Wikipedia.

The next row, 'Base features', shows that lexical features are always better than wikifier features only. This agrees with the common wisdom that lexical features are important for NER.

Adding gazetteers to the base features improves by more than 3 points for higher-resource languages. This is because the low-resource languages have much smaller gazetteers which have lower coverage than other languages' gazetteers.

[1]We set n to 4 in all our experiments.

Finally, the '+Wikifier' row shows that our proposed features are valuable even in combination with strong features. It improves upon base features and gazetteer features for all 9 languages. These numbers may be less than state of the art because the features we use are designed for English, and may not capture lexical subtleties in every language. Nevertheless, they show that wikifier features have a non-trivial signal that has not been captured by other features.

Direct Transfer Experiments

We evaluate our direct transfer experiments by training on English and testing on the target language. The results from these experiments are shown in the bottom section of Table 12.

The 'Wikifier only' row shows that the wikifier features alone preserve a signal across languages. Interestingly, for both Bengali and Tamil, this is the strongest signal, and gets the highest score. If the lexical features are included when we train the English model, the learning algorithm will give them too much emphasis, thus decreasing the importance of the wikifier features. Since Bengali and Tamil use non-Latin scripts, no lexical feature in English will fire at test time. Thus, approaches that include base features perform poorly.

The results of 'Base features' can be viewed as a sort of language similarity to English, which, in this case, is related to lexical overlap and similarity between the scripts. Comparing to monolingual experiments, we can see that the lexical features become weak in the cross-lingual setting.

The gazetteer features are again shown to be very useful for almost all languages except Bengali and Tamil due to the reason explained in the monolingual experiment and to the inclusion of lexical features. For all other languages, the gain from adding gazetteers is even larger than it is in the monolingual setting.

For nearly every language, wikifier features help dramatically, which indicates that they are very good delexicalized features. Wikifier features add more than 10 points on Dutch, German, and Turkish.

The trend in Table 12 suggests the following strategy when we want to extract named entities in a new foreign language: It is better to include all features if the foreign language uses Latin script, since the names are likely to be mentioned similarly to the English names. Otherwise, using wikifier features only could be the best setting.

Täckström et al. (2012) also directly transfer an English NER model using the same setting as ours: train on the CoNLL English training set and predict on the test set of other three languages. We compare our baseline transfer model (Base Features) to the row denoted by "Täckström baseline". Even though we do not use gold POS tags, we see that our results are comparable. The second Täckström row uses parallel text to induce multilingual word clustering. While this approach is orthogonal to ours, and could be used in tandem to get even better scores, we compare against it for lack of a more closely aligned scenario. We see that for each language, our approach significantly outperforms their approach.

We note that our numbers are comparable to those reported for WIKI-2 in (Nothman et al., 2013) for the CoNLL languages (with the exception of German, where their result is higher). However, they require language-specific heuristics to generate *silver-standard* training data from Wikipedia articles. What they gain for single languages, they likely lose in generalization to other languages. This approach is orthogonal to ours; we, too, can use their silver-standard data in training.

For the low-resource languages, we compare our direct transfer model with the expectation learning model proposed in (Zhang et al., 2016). This model is not a direct transfer model, but it does not use any training data in the target languages either. Instead, for each target language, it generates patterns from parallel documents between English and the target language, a large monolingual corpus in the target language, and one-hour interaction with a native speaker of the target language. Note that they also use a cross-lingual wikifier, but only for refining the entity types. On the other hand, in our model, the features from the

TRAINING LANG	TR	TL	YO	AVG
EN	47.12	65.44	36.65	49.74
EN+ES	44.85	66.61	37.57	49.68
EN+NL	48.34	66.09	36.87	50.43
EN+DE	49.47	64.10	35.14	49.57
EN+ES+NL+DE	49.00	66.37	**38.02**	51.13
ALL−Test Lang	**49.83**	**67.12**	37.56	**51.50**

Table 11: The F1 scores of the proposed direct transfer model on three low-resource languages using training data in multiple languages. The row "ALL−Test Lang" trains the model on all languages except the test language, Bengali, and Tamil. Bengali and Tamil are excluded since we use all features in this experiment.

wikifier are used both in detecting entity mention boundaries and entity types. We can see that our approach performs better than their model on all five languages even though we assume much fewer resources. The difference is most significant on Turkish, Tagalog, and Bengali.

Training Languages

In all previous experiments, the training language is always English. In order to test the efficacy of training with languages other than English, we created a train/test matrix with all combinations of languages.

Table 11 shows the results of training on multiple languages. We use all features in this experiment. The row "EN" only trains the model on the English training documents, and the results are identical to those shown in Table 12. Using all CoNLL languages (EN+ES+NL+DE) adds more than 1 point F1 in average comparing to using English only. Finally, training on all but the test languages further improves the results.

This experiment shows that we can augment training data from other languages' annotated documents. Although the performance only increases a little, it does not hurt most of the time.

4.1.3. Conclusion and Discussion

We propose a language-independent model for cross-lingual NER building on a cross-lingual wikifier. This model works on all languages in Wikipedia and the only requirement is a Wikipedia dump. We study a wide range of languages in both the monolingual and the cross-lingual settings, and show significant improvements over strong baselines. An analysis shows that the quality of the wikifier features depends on the Wikipedia size of the test language.

This work shows that if we can disambiguate words and phrases to the English Wikipedia, the typing information from Wikipedia categories and FreeBase are useful language-independent features for NER. However, there is additional information in Wikipedia that could be helpful and which we do not use, including words in the documents and relations between titles; this would require additional research.

4.2. Cross-lingual NER via Cheap Translation

This work published as (Mayhew et al., 2017b).

We continue with the task of low-resource NER. The previous chapter showed how to use Wikipedia in low-resource settings. But while Wikipedia extends to nearly 300 languages, there are still many languages not in Wikipedia, and many of those present are too small to be useful. Parallel text may be found on an ad-hoc basis for some languages, but it is hardly a general solution. Religious texts, such as the Bible and the Koran, exist in many languages, but the unique domain makes them hard to use. This leaves the vast majority of the world's languages with no general method for NER.

We propose a simple solution that requires only minimal resources. We translate annotated data in a high-resource language into a low-resource language, using just a lexicon.[2] We refer to this as *cheap translation*, because in general, lexicons are much cheaper and easier

[2]We use the terms 'lexicon' and 'dictionary' interchangeably.

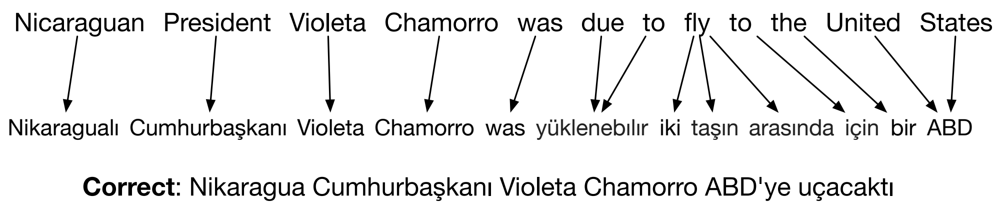

Nicaraguan President Violeta Chamorro was due to fly to the United States
Nikaragualı Cumhurbaşkanı Violeta Chamorro was yüklenebılır iki taşın arasında için bir ABD
Correct: Nikaragua Cumhurbaşkanı Violeta Chamorro ABD'ye uçacaktı

Figure 14: Demonstration of word translation. The top is English, the bottom is Turkish. Lines represent dictionary translations (e.g. *the* translates to *bir*). **Correct** is the correct translation. This illustrates congruence in named entity patterns between languages, as well as some errors we are prone to make.

to find than parallel text (Mausam et al., 2010).

We show that our approach gives non-trivial scores across several languages, and when combined with orthogonal features from Wikipedia, improves on state-of-the-art scores.

4.2.1. Cheap Translation

We create target language training data by translating source data into the target language. It is effectively the same as standard phrase-based statistical machine translation systems (such as MOSES (Koehn et al., 2007)), except that the translation table is not induced from expensive parallel text, but is built from a lexicon, hence the name *cheap translation*.

The entries in our lexicon contain word-to-word translations, as well as word-to-phrase, phrase-to-word, and phrase-to-phrase translations. A standard problem related to ambiguity in language applies: a source language word may have several translations.

We addressed this problem by ranking translation pairs according to cooccurrence counts of tokens in the dictionary. For example, in one English-Spanish lexicon, the English word *woman* translates into about 50 different Spanish words, with meanings ranging from *woman*, to *female golfer*, to *youth*, but the token *woman* occurs most frequently with the Spanish token *mujer*. We normalize these cooccurrence counts in each candidate set, and call this the *prominence score*.

With these probabilities in hand, we have effectively constructed a phrase translation table.

We use a simple greedy decoding method where options from the lexicon are resolved by a language model multiplied by the prominence score of each option.

During decoding, once we have chosen a candidate, we copy all labels from the source phrase to the target phrase. Since the translation is phrase-to-phrase, we can copy gold labels directly,[3] without worrying about getting good alignments. The result is annotated data in the target language.

Notice that the algorithm allows for no reordering beyond what exists in the phrase-to-phrase entries of the lexicon. Compared to phrase-tables learned from massive parallel corpora, our lexicon-based phrase tables are not large enough or expressive enough for robust reordering. We leave explorations of reordering to future work.

See Figure 14 for a representative example of translation from English to Turkish, with a human translation as reference. There are single words translated into phrases, named entities copied over verbatim, and phrases translated into single words. Some words are translated correctly (*President* into *Cumhurbaşkanı*) and some incorrectly (*fly* into *iki taşın arasında*, which loosely translates to 'between two stones'). We see ignorance of morphology (seen in translation of United States), and confused word order. But in spite of all these mistakes, the context around the entities, which is what matters for NER, is reasonably well-preserved. Notably, the word *President/Cumhurbaşkanı* is a strong context feature for both LOC (Nicaragua) and PER (Violeta Chamorro) in both languages.

4.2.2. Experimental Setup

Before we describe our experiments, we describe some of the tools we used. We use lexicons provided by (Rolston and Kirchhoff, 2016), which are harvested from PanLex, Wiktionary, and various other sources. There are 103 lexicons, each mapping between English and a target language. These vary in size from 56K entries to 1.36M entries, as shown in the second row of Table 12.

[3]We use a standard BIO labeling scheme.

As before, we use data from CoNLL2002/2003 shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003), and also several LORELEI language packs. We use the same set of test documents as used in the previous chapter.

We use the Illinois NER system (Ratinov and Roth, 2009) with standard features (forms, capitalization, affixes, word prior, word after, etc.) as our base model. We train Brown clusters on the entire Wikipedia dump for any given language (again, any monolingual corpus will do), and include the multilingual gazetteers and wikifier features proposed in Section 4.1.

4.2.3. Experiments

We performed two different sets of experiments: first translating only from English, then translating from additional languages selected to be similar to the target language.

Translation from English

We start by translating from the highest resourced language in the world, English. We first show that our technique gives large improvement over a simple baseline, then combine with orthogonal features, then compare against a ceiling obtained with Google Translate.

Baseline Improvement

To get the baseline, we trained a model on English CoNLL data (train set), and applied the model directly to the target language, mismatching lexical features notwithstanding. We did not use gazetteers in this approach. For the non-Latin script languages, Tamil and Bengali, we transliterated the entire English corpus into the target script. These results are in Table 12, row "Baseline".

In our approach ("Cheap Translation"), for each test language, we translated the English CoNLL data (train set) into that language. The first row of Table 12 shows the coverage of each dictionary. For example, in the case of Spanish, 90.94% of the words were translated

Method	Dutch	German	Spanish	Turkish	Bengali	Tamil	Yoruba	Avg
Lexicon Coverage	88.01	89.97	90.94	83.80	83.34	73.84	74.60	–
E-L Dict size	961K	1.36M	1.25M	578K	217K	182K	334K	–
Baseline	43.10	22.61	45.77	34.63	6.40	4.60	37.70	27.83
Google Translate ceiling	65.71	56.65	53.65	45.63	37.84	29.11	39.18	46.82
Wiki (Tsai et al., 2016)	61.56	48.12	60.55	47.12	43.27	29.64	36.65	46.70
Cheap Translation	53.94	50.96	51.82	46.37	30.47†	25.91†	37.58	42.43
Cheap Translation+Wiki	63.37	57.23	64.10	51.79	**46.28**†	33.10†	38.52	50.62
Best Combination	64.48	57.53	**65.95**	48.50	31.70†	27.63†	39.12	47.84
Best Combination+Wiki	**66.50**	**59.11**	65.43	**53.44**	45.70†	**34.90**†	**40.88**	**52.28**

Table 12: Baseline is naive direct transfer, with no gazetteers. 'Cheap Translation' translates from English into the target. Google Translate translates whole sentences, and does not use gazetteers. 'Cheap Translation+Wiki' incorporates wikifier features. 'Best Combination' uses language combinations for source training data. † denotes that this run does not use word features.

into Spanish. This gives an average of 14.6 points F1 improvement over the baseline. This shows that simple translation is surprisingly effective across the board. The improvement is most noticeable for Bengali and Tamil, which are languages with non-Latin script. This mostly shows that the trivial baseline doesn't work across scripts, even with transliteration. Spanish shows the least improvement over the baseline, which may be because English and Spanish are so similar that the baseline is already high.

We found that we needed to normalize the Yoruba text (that is, remove all pronunciation symbols on vowels) in order to make the data less sparse. Since the training data for Bengali and Tamil never shares a script with the test data, we omit using the word surface form as a feature. This is indicated by the † in Table 12. Brown clusters, which implicitly use the word form, are still used.

Wikifier Features

Our approach is orthogonal to the work in the Chapter 4.1, and thus can be naturally combined with wikifier features. We show results in Table 12, in the row marked 'Cheap Translation+Wiki'.

Using wikifier features improves scores for all 7 languages. Further, for all languages we beat the prior scores with an average of 3.92 points F1 improvement. For the three European languages (Dutch, German, and Spanish), we have an average improvement of 4.8 points F1 over (Tsai et al., 2016). This may reflect the fact that English is more closely related to European languages than Indian or African languages, in terms of lexical similarities, word order, and spellings and distribution of named entities. This suggests that it is advantageous to select a source language similar to the target language (by some definition of similar). We explore this hypothesis in Section 4.2.3.

Google Translate

To get an idea of a ceiling in performance, we used Google Translate to translate the English CoNLL training data into the target language, sentence by sentence. We aligned the source-target data using fast align (Dyer et al., 2013), and projected labels across alignments.[4] As with the other approaches, we found that Brown cluster features were an important signal.

Surprisingly, Google Translate beats our basic approach with a margin of only 4.3 points. Despite the naïvete of our approach, we are relatively close to the ceiling. Further, Google Translate is limited to 103 languages, whereas our approach is limited only by available dictionaries. In low-resource settings, such as the one presented in Section 4.2.4, Google Translate is not available, but dictionaries are available, although perhaps only by pivoting through a high-resource language.

Translation from Similar Languages

Observing that English as a source works well for European languages, but not as well for non-European languages, we form a key hypothesis: *cheap translation between similar languages should be better than between different languages.* There are several reasons for

[4]Google Translate does not output alignments. If we had an in-house translation system, we could avoid this step.

this. First, similar languages should have similar word orderings. Since we do no reordering in translation, this means the target text has a better chance of a coherent ordering. Second, in case of dictionary misses, vocabulary common between languages will be correct in the target language.

This requires two new resources: annotated data in a similar language S, and a lexicon that maps from S to T, the target language.

Data in other languages

For most target languages, English is not the closest language, and it is likely that there exists an annotated dataset in a closer language. There are annotated datasets available in many languages with a diversity of script and family. We have datasets annotated in about 10 different languages, although more exist. The only caveat is that the source dataset must have a matching tagset with the target dataset.

Pivoting Lexicons

Although we cannot expect to find lexicons between all pairs of languages, we can usually expect that a language will have at least one lexicon with a high-resource language, which is usually English. We can use this high-resource language as a pivot to transitively create an S–T dictionary, although perhaps with some loss of precision.

Notice that the resulting dictionary contains only those English words that appear in both original dictionaries. If either of the original dictionaries is small, the result will be smaller still.

Source Selection and Combination

We chose related languages automatically, using syntactic features of languages retrieved from the World Atlas of Language Structures (WALS) (Dryer and Haspelmath, 2013). We used the feature set called *syntax_knn*, which includes 103 syntactic features, such as *subject before verb*, and *possessive suffix*, and uses k-Nearest Neighbors to predict missing values. We measure similarity as cosine distance between language vectors.

In the absence of criteria for a similarity cutoff, we chose to include only the top most similar language as source for that target language. The results of this similarity calculation are always sensible. For example, when the target language is Dutch, German is the closest. We also included English in the training, as the highest resource language, and with the highest quality dictionaries.

Results

Our results are in Table 12, in the row named 'Best Combination'. The average over all languages surpasses the English-source average by 5.4 points, and also beats (Tsai et al., 2016). We also add wikifier features, and report results in row 'Best Combination+Wiki.' This shows improvement on all but Spanish, with an average improvement of 5.58 points F1 over (Tsai et al., 2016). To the best of our knowledge, these are the best cross-language settings scores for all these datasets.

While these scores are lower than those seen on typical NER tasks (70-90% F1), we emphasize first that cross-lingual scores will necessarily be much lower than monolingual scores, and second that these are the best available given the setup.

4.2.4. Case study: Uyghur

We have shown in the previous sections that our method is effective across a variety of languages. However, all of the tested languages have some resources, most notably, Google Translate and reasonably sized Wikipedias. In this section, we show that our methods hold up on a truly low-resource language, Uyhgur.

Uyghur is a language native to northwest China, with about 25 million speakers.[5] It is a

[5]`https://en.wikipedia.org/wiki/Uyghur_language`, accessed May, 2018

Turkic language, and is related most closely to Uzbek, although it uses an Arabic writing system. Uyghur is not supported by Google Translate, and the Uyghur Wikipedia has less than 3,000 articles. In contrast, the smallest Wikipedia size language in our test set is Yoruba, with 30K articles. Because of the small Wikipedia size, we do not use any wikifier features.

We did this work as part of the NIST LoReHLT evaluation in the summer of 2016. The official evaluation scores were calculated over a set of 4500 Uyghur documents. Each team was given the unannotated version of those documents, with the task being to submit annotations on that set. After the evaluation, NIST released 199 of the annotated evaluation documents, called the *unsequestered set*. In this section, we will drill into the various methods we used to build the transfer model, and report finer-grained results using the unsequestered set.

The following are some of the language-specific techniques we employed.

- **Dictionary** The dictionary provided for Uyghur from (Rolston and Kirchhoff, 2016) had only 5K entries, so we augmented this with the dictionary provided in the LORELEI evaluation, which resulted in 116K entries.

- **Name Substitution** As with Bengali and Tamil, very few names were translated. We found transliteration models were too noisy, so instead, we gathered a list of gazetteers from Uyghur Wikipedia, categorized by tag type (PER, LOC, GPE, ORG). Upon encountering an untranslatable NE, we replaced it with a randomly selected NE from the gazetteer list corresponding to the tag. This led to improbable sentences like *John Kerry has joined the Baskin Robbins*, but it meant that NEs were fluent in the target text.

- **Stemming** We created a very simple stemmer for Uyghur. This consists of 45 common suffixes sorted longest first. For each Uyghur word in a corpus, we removed all possible

Method	F1
Monolingual	69.92
Standard Translation	
Train: English	27.20
Train: Turkish	33.02
Train: Uzbek	27.88
Language Specific (stemmed)	
Train: English	30.84
Train: Turkish	40.04
Train: Uzbek	40.15
Induced dictionaries	43.46
Manual annotations	42.51
All Lang. Spec.	51.32

Table 13: F1 scores for Uyghur. Monolingual scores are on the 41 document test set. All other scores are on the full unsequestered data. We omit forms or gazetteers but use Brown clusters. 'Standard Translation' uses the same resources as the scores in Table 12 (e.g. without stemming)

suffixes (Uyghur words can take multiple suffixes). We stemmed all train and test data.

We report results in Table 13. The first row is from a monolingual model trained on 158 documents in the unsequestered set, and tested on the remaining 41. All other rows test on the complete unsequestered set. The next section, 'Standard Translation', refers to the method described above. Notably, we do not use stemming for train or test data here. As with Bengali and Tamil, we omit form features.

We translate from English, Turkish, and Uzbek, which are the closest languages predicted by WALS. Next, we incorporated language specific methods. The scores we get from training on English, Turkish and Uzbek all go up because the stemming makes the features more dense. Next we generated dictionaries using observations over Uyghur and Uzbek, and we used non-native speakers to annotate Uyghur data.

Language Specific Dictionary Induction

We began by romanizing Uyghur text into the Uyghur Latin alphabet (ULY) so we could read it. We noticed that Uzbek and Uyghur are very similar, sharing a sizable amount of vocabulary, and several morphological rules. However, while there is a shared vocabulary, the words are usually spelled slightly differently. For example, the word for "southern" is "janubiy" in Uzbek and "jenubiy" in Uyghur.

We tried several ideas for gathering a mapping for this shared vocabulary: manual mapping, edit-distance mapping, and cross-lingual CCA with word vectors.

Manual mapping: We manually translated about 100 words often found around entities, such as *president*, and *university*

Edit-distance mapping: We gathered (Uyghur, Uzbek) word pairs with low-edit distance, using a modified edit-distance algorithm that allowed certain substitutions at zero cost. For example, this discovered such pairs as *pokistan-pakistan* and *telegraph-télégraf.*

Cross-lingual CCA with word vectors: We projected Uyghur and Uzbek monolingual vectors into a shared semantic space, using CCA (Faruqui and Dyer, 2014). We used the list of low edit-distance word pairs as the dictionary for the projection. Once all the vectors were in the same space, we found the closest Uyghur word to each Uzbek word.

Results

Scores are in Table 13. Interestingly, the language specific methods evaluated individually did not improve much over the generic word translation methods. But with all language specific methods combined, 'All Lang. Spec.', the score increased by nearly 10 points, suggesting that the different training data covers many angles.

To the best of our knowledge, there are no published scores on the unsequestered data set. Our best score is comparable to the score of our evaluation submission on the unsequestered dataset.

4.2.5. Conclusion

We have shown a novel cross-lingual method for generating NER data that gives significant improvement over state-of-the-art on standard datasets. The method benefits from annotated data in many languages, combines well with orthogonal features, and works even when resources are virtually nil. The simplicity and minimal use of resources makes this approach more portable than all previous approaches.

CHAPTER 5 : Massively Multilingual Analysis of NER

5.1. Introduction

The past several years have brought increased interest in low-resource techniques for NER, resulting in several new works, many of which have been described in this dissertation. Although new works tend to compare against prior methods, it's difficult to maintain a comprehensive picture of which techniques work well, and how they compare to each other. Models and parameters differ, and choice of languages for evaluation is often *ad hoc*, leading to a patchwork understanding of the state of the art.

In this chapter, we take stock of the field of low-resource NER by choosing several representative techniques from the literature, and applying them to a massively multilingual corpus of over 20 languages (Strassel and Tracey, 2016). By keeping a consistent process, we are able to measure differences in performance between methods, and across languages, thereby painting an accurate picture of the current multilingual NER landscape.

In all of our cross-lingual experiments, we use an English annotated dataset as the source of supervision. The techniques we experiment with run the gamut from direct transfer from English, transfer over cross-lingual embeddings (created with and without bilingual dictionaries), cheap translation of the English data into the target language, and transfer over multilingual contextual embeddings. In addition to the cross-lingual experiments, we include monolingual (that is, non-cross-lingual) runs with gold training data to establish an upper bound.

We show that recent methods using bilingual lexicons significantly outperform cross-lingual embedding transfer methods. Further, we confirm recent work (Wu and Dredze, 2019) that shows surprisingly strong cross-lingual performance from a multilingual version of BERT (Devlin et al., 2019), even with no cross-lingual resources.

5.2. Methods

Our experiments are divided into monolingual methods, which use training data in the target language, and cross-lingual, or zero-shot methods, which use no training data in the target langauge. Before diving into the details of the experiments, we give an overview of each subset.

5.2.1. Monolingual

In our monolingual experiments, we train on gold training data, with different configurations of models and embeddings.

First, we ran Cogcomp NER (Ratinov and Roth, 2009), using out of the box parameters, and with Brown clusters trained over large monolingual text. This model is based on averaged Perceptron, and uses handcrafted features. Although this model has been used for cross-lingual methods in the past (Tsai et al., 2016; Mayhew et al., 2017b), we use it only for monolingual NER, leaving the cross-lingual methods for the neural model for the sake of fair comparison.

For a neural model, we used the standard BiLSTM-CRF model from AllenNLP (Gardner et al., 2018) with pre-trained word embeddings (either static or contextual).

5.2.2. Cross-lingual

We experiment with several different cross-lingual methods, each of which we describe briefly here. In each model, we use the standard BiLSTM-CRF, differing only in training data, embeddings, and in one case, adding a self-attention layer.

1. **Cross-lingual embeddings only** In these experiments, we follow prior work on cross-lingual embeddings, in which two monolingual embedding spaces are mapped together to form a shared semantic space across languages. We start with the fastText

Wiki pretrained embeddings, and use MUSE (Conneau et al., 2018) code,[1] to generate cross-lingual embeddings supervised with bilingual word mappings. We use two methods to gather these mappings: from identical surfaces shared between languages (often names or numbers), and from bilingual dictionaries. We always transform the foreign language embeddings into English, which allows us to train a single English NER model using the English embeddings. Our evaluation on NER follows in the spirit of recent work on using extrinsic tasks to evaluate bilingual word embeddings (Glavaš et al., 2019),

2. **Cheap Translation** Following after Mayhew et al. (2017b), we translate English training data word for word into the target language using the dictionaries described above. If a word isn't translated, it is simply copied over. Then, with training data in the target language, we train a model using monolingual embeddings, exactly as though we were learning a monolingual model from gold standard data. We also experiment with the follow-up work from Xie et al. (2018) in which the dictionaries are gathered using (supervised) cross-lingual embeddings (Conneau et al., 2018), and a self-attention layer (Vaswani et al., 2017) is added. For all of these experiments, we use the code from Xie et al. (2018).[2]
3. **Cross-lingual Contextual Embeddings** We experiment with cross-lingual contextual embeddings (Devlin et al., 2019), after they have been shown to be surprisingly effective despite using no cross-lingual resources (Wu and Dredze, 2019). The training paradigm is the same as for cross-lingual embeddings: we train a model on English data using cross-lingual contextual embeddings, and evaluate the model directly to the target language.

[1]`https://github.com/facebookresearch/MUSE`

[2]`https://github.com/thespectrewithin/cross-lingual_NER`

Language	Code	Mono ID	Anno ID
Akan (Twi)	aka	LDC2018E06	LDC2018E07
Amharic	amh	LDC2016E86	LDC2016E87
Arabic	ara	LDC2016E88	LDC2016E89
Bengali	ben	LDC2017E59	LDC2017E60
Farsi	fas	LDC2016E92	LDC2016E93
Hindi	hin	LDC2017E61	LDC2017E62
Hungarian	hun	LDC2016E98	LDC2016E99
Indonesian	ind	LDC2017E65	LDC2017E66
Mandarin	cmn	LDC2016E100	LDD2016E101
Russian	rus	LDC2016E94	LDC2016E95
Somali	som	LDC2016E90	LDC2016E91
Spanish	spa	LDC2016E96	LDC2016E97
Swahili	swa	LDC2017E63	LDC2017E64
Tagalog	tgl	LDC2017E67	LDC2017E68
Tamil	tam	LDC2017E69	LDC2017E70
Thai	tha	LDC2018E02	LDC2018E03
Turkish	tur	LDC2014E115	–
Uzbek	uzb	LDC2016E29	–
Vietnamese	vie	LDC2016E102	LDC2016E103
Wolof	wol	LDC2018E08	LDC2018E09
Yoruba	yor	LDC2016E104	LDC2016E105
Zulu	zul	LDC2018E04	LDC2018E05

Table 14: Language codes and dataset IDs. Each corpus (with 3 exceptions) is comprised of a monolingual corpus, and a set of annotations that describe a subset of the monolingual text.

5.3. Experimental Setup

5.3.1. Data

We use the LORELEI Resource Language Packs (Strassel and Tracey, 2016), hereafter referred to as LRLPs, as described in Section 2.1.2, with language codes and dataset ID shown in Table 14. We define train/test splits for all languages, careful to split evenly each genre represented (newswire, social network, weblog, discussion forum, etc). See Table 15 for details on the sizes of each dataset and split.

When

	Train		Test		
Language	docs	tokens	docs	tokens	Dictionary Size
aka	276	67,343	70	17,502	6,124
amh	414	48,221	106	14,140	19,501
ara	377	55,881	96	14,302	338,011
ben	887	110,714	224	31,100	143,485
cmn	258	93,984	66	23,035	993,851
fas	309	49,090	81	12,342	165,318
hin	395	71,568	101	19,619	208,365
hun	369	56,735	94	14,747	960,428
ind	482	76,084	123	16,177	304,334
rus	384	57,328	99	15,816	628,473
som	447	51,844	114	14,066	30,051
spa	256	47,092	65	8,742	721,624
swa	573	72,404	145	19,448	94,278
tam	674	111,846	171	28,369	223,121
tgl	412	101,798	107	29,519	91,590
tha	555	191,547	142	48,543	591,255
tur	572	61,856	143	18,993	356,245
uzb	380	126,641	96	24,206	47,716
vie	326	50,474	83	14,651	117,035
wol	293	67,281	74	14,606	2,538
yor	227	57,299	58	14,250	53,843
zul	1601	74,434	402	17,985	17,852

Table 15: Train/Test splits for each language

	Train		Dev		Test	
	Docs	Tokens	Docs	Tokens	Docs	Tokens
CoNLL	537	116,284	38	9,274	46	9,213
LRLP	109	17,568	30	5,417	31	4,507
Total	646	133,852	68	14,691	77	13,720

Table 16: Statistics for English training data.

5.3.2. Resources

Brown Clusters

We trained Brown clusters for use in the Cogcomp NLP system. For each language, we used the entire monolingual corpus provided in the LRLP.[3] For all languages, we used minimum word threshold of 3, and cluster size of 2000, in response to the polite request in Derczynski et al. (2015).

Word Embeddings

In keeping with recent work on bilingual word embeddings (Conneau et al., 2018; Glavaš et al., 2019), we used the 300 dimensional pretrained fastText embeddings trained over Wikipedia.[4] According to the website, these vectors "were obtained using the skip-gram model described in (Bojanowski et al., 2017) with default parameters." These embeddings are all lower-cased. FastText also provides pretrained embeddings trained over Common Crawl plus Wikipedia, but not all LRLPs are represented in that set.

For contextual embeddings, we use multilingual BERT (Devlin et al., 2019), hereafter mBERT. mBERT is trained over the top 104 languages in Wikipedia, but excludes Chinese, which is released as a separate model. Not all languages or scripts from the LRLPs are present in Wikipedia data used to train mBERT. In particular, the missing languages are: Akan, Amharic, Wolof, Zulu. We would thus expect performance on these 4 languages (and Chinese) to be relatively poor.

Lexicons

Several of the techniques presented here use bilingual lexicons. Although good-quality lexicons are released as part of the MUSE project (Conneau et al., 2018), these do not cover

[3]We used `https://github.com/percyliang/brown-cluster`.

[4]`https://fasttext.cc/docs/en/pretrained-vectors.html`

all the languages in the LRLPs. Instad, we use lexicons from the Masterlex project (Rolston and Kirchhoff, 2016), which include hundreds of languages, and full coverage of our target languages.

These lexicons are automatically harvested, and are quite noisy, with many repeated or incorrect entries, and many entries that are phrases. Since these phrase-entries are not useful for word embedding purposes, we preprocessed each lexicon to include only entries that consist of single word pairs. The number of entries in each dictionary per language is shown in Table 15.

5.4. Monolingual Results

First, we show results from models trained on gold annotated text in Table 17. These experiments should serve as an upper bound to later cross-lingual experiments. These may also serve as a baseline for future experimentation on these languages.

At a first glance, it's interesting to see that the absolute values are relatively low. We are used to scores in the 90s for English CoNLL, and in the high 80s for the other CoNLL languages, and OntoNotes. The average of these datasets is around 74 F1.

When comparing the different methods, the average row at the bottom of the table shows some surprising results. The CCG results are surprisingly strong compared to the neural network results, although the mBERT experiments are the strongest overall. This may be because of the difference in text used to train the different representations: Wikipedia for the embeddings and mBERT, but the LRLP text for the Brown clusters. Since both mBERT and the embeddings were trained on Wikipedia text, we can compare them directly. However, this difference may also be explained by the fact that the wordpiece vocabulary of mBERT results in very few unknown tokens (which are often names anyway), whereas the fastText embeddings trained on Wikipedia may have many out of vocabulary words in the LRLPs.

Language	CCG	fastText Wiki	mBERT
aka	72.46	68.87	**76.37**
amh	**69.15**	60.74	52.01
ara	55.22	54.21	**61.43**
ben	**77.10**	73.48	76.50
cmn	**74.18**	70.06	68.93
fas	61.72	60.50	**66.36**
hin	71.63	68.69	**72.27**
hun	62.80	59.29	**78.30**
ind	66.64	56.80	**75.57**
rus	67.18	66.07	**77.35**
som	76.75	70.93	**77.37**
spa	67.50	58.51	**76.37**
swa	**76.77**	70.34	76.35
tam	62.85	66.80	**69.55**
tgl	81.61	76.35	**86.65**
tha	73.49	**77.29**	75.76
tur	76.33	69.14	**84.97**
uzb	78.17	73.46	**80.87**
vie	56.49	51.37	**68.72**
wol	**76.57**	71.38	73.68
yor	73.08	66.23	**73.84**
zul	75.64	78.32	**81.55**
avg	70.61	66.77	**74.13**

Table 17: Results from the Monolingual Experiments

		Identical Chars			Dictionary		
Language	Latin?	char	nochar	By script	char	nochar	By script
aka	yes†	**21.67**	2.18	21.67	15.41	0.00	15.41
amh	no	2.74	2.18	2.18	3.90	**11.67**	11.67
ara	no	0.59	**28.98**	28.98	0.59	24.75	24.75
ben	no	1.91	8.80	8.80	5.43	**40.37**	40.37
cmn	no	0.00	**0.74**	0.74	0.00	0.48	0.48
fas	no	1.72	38.21	38.21	1.71	**39.13**	39.13
hin	no	3.28	20.22	20.22	4.72	**34.11**	34.11
hun	yes*	32.97	**47.22**	32.97	32.57	46.53	32.57
ind	yes	53.44	46.48	53.44	**53.99**	43.33	53.99
rus	no	10.12	46.27	46.27	9.00	**47.41**	47.41
som	yes	24.80	4.15	24.80	**29.59**	16.02	29.59
spa	yes	54.29	**61.59**	54.29	54.48	58.61	54.48
swa	yes	35.70	9.84	35.70	**47.95**	27.86	47.95
tam	no	2.13	18.97	18.97	3.99	**25.96**	25.96
tgl	yes	60.83	48.21	60.83	**62.71**	44.35	62.71
tha	no	2.46	28.67	28.67	3.39	**31.82**	31.82
tur	yes†	**48.27**	39.11	48.27	48.01	38.47	48.01
uzb	yes*	45.57	28.79	45.57	**47.30**	30.91	47.30
vie	yes*	35.07	27.32	35.07	**35.37**	30.86	35.37
wol	yes*	12.29	0.70	12.29	**15.28**	6.25	15.28
yor	yes*	26.36	2.51	26.36	**30.37**	12.98	30.37
zul	yes	14.30	1.59	14.30	**16.49**	1.44	16.49
avg		21.33	22.29	28.64	22.71	26.67	**32.40**

Table 18: Results from the Cross-lingual Embeddings Experiments. †with added characters. *with diacritics.

5.5. Cross-lingual Experiments

Having gathered scores from supervised gold annotations, we now move to the low-resource cross-lingual settings. In all cases, we use English training data. We use only neural methods in these experiments, for the sake of consistency across approaches, and because most recent cross-lingual methods use resources such as word embeddings that are hard to use in non-neural contexts.

5.5.1. Results

We show results in two stages. First, we will show results from the different variations of cross-lingual embeddings, then combine all methods to show final results.

Cross-lingual Embeddings All variations of cross-lingual embeddings are shown in Table 18. The first column in this table indicates if the language in each row uses Latin script. In some cases, such as Hindi (hin), it clearly uses a separate script, but in other cases, such as Akan (aka) and Hungarian (hun), the picture is a little more complicated, using either extra characters (marked with †) or using many diacritics (marked with *). For example, Akan uses an alphabet with two characters not found in English: ɛ and ɔ. Although these are only two characters, they are ubiquitous in Akan writing, as seen in an example sentence: Yei ɛdane butuiɛ a anka nkɔnkɔnsa. Hungarian, on the other hand, has a large number of diacritics, for example: Felügyeleti felszólítás a MÁV Biztosítónak Két határozatot.

The next four columns contain combinations of identical character vs dictionary embedding approach, and whether the NER model used character embeddings during training (char) or not (nochar). We may think of character embeddings as being a kind of lexicalization of the word embeddings, and we may expect that a language with Latin script would benefit from using character embeddings because of the likelihood of shared entity surface forms (which is to say that the entity will not be transliterated). On the other hand, for languages that are written in non-Latin scripts, these character features will likely be harmful, as the model will fit closely to character sequences that it never sees at test time.

The *By script* columns select scores according to these hypotheses. If a language uses Latin script, it selects the entry from the *char* column. Otherwise it selects the score from the *nochar* column.

Looking only at average scores in the bottom row of Table 18, we see that the dictionary-based embeddings outperform the unsupervised embeddings by nearly 4 points F1. This is unsurprising given the resources used. As we look to understand the other results, we

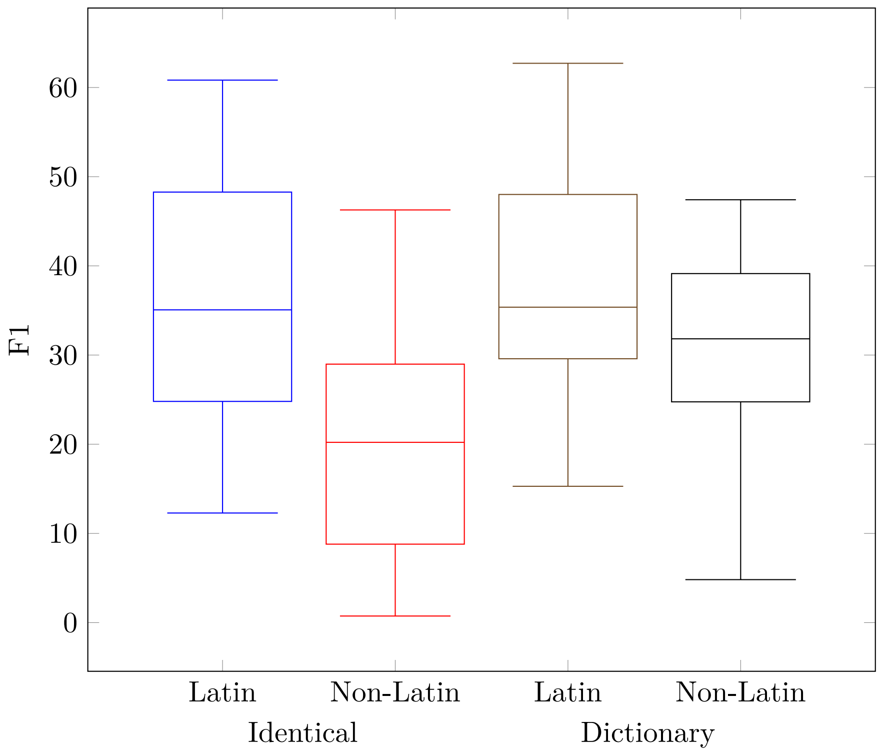


Figure 15: Box plots for languages using Latin vs non-Latin scripts in both identical characters and dictionary initializations. As one might expect, the use of the dictionary is most helpful to the languages with non-Latin script.

consider three factors: 1) the quality of the mapping, 2) the quality of the embeddings, and 3) the overlap in surface forms. At test time, we always used only embeddings from target language (e.g. none from English), there is no notion of direct transfer benefit from shared vocabulary. For the identical character experiments, factors 1) and 3) are closely related.

We examine the identical character experiments first. As expected, languages with non-Latin scripts have significantly lower scores than those with Latin scripts. This is likely due to the first factor: since there is little shared vocabulary, the mapping is weak. For languages with Latin script, having character embeddings can be a saving grace, as with Akan, Wolof, Yoruba, and Zulu.

Certain non-Latin script languages have quite high performance, such as Russian and Farsi. These were ranked 7th and 18th respectively in Wikipedia size as of writing, and 8382 and

15059 pairs of identical strings during the embedding mapping process. These both point to higher quality embeddings. A larger amount of text for training means better embeddings, but also suggests that there are more English words in the embedding vocabulary.

When adding a dictionary, most languages improve. In particular, Bengali and Amharic (both non-Latin script) are able to get some purchase where before they floundered. In one peculiar case, Chinese (cmn), the scores never reach reasonable values. To the best of my knowledge, this is because of unusually poor embeddings. The cause of this is not clear, but may have to do with the preprocessing of Chinese Wikipedia, perhaps using non-standard segmentation. In another peculiar case, Akan, the dictionary actually harms the scores, perhaps also because of poor dictionary quality. Figure 15 shows box plots comparing statistics of Latin and Non-Latin script languages in both the identical character and dictionary scenarios. Interestingly, it is the languages with non-Latin scripts that improve the most with the addition of the dictionary.

Given the high performance, in later sections we report only the *By script* column from the dictionary experiments.

While these experiments are interesting at first glance, there remain several open questions. We have used dictionaries of varying sizes and unknown quality. It is not clear how much these factors affect performance. Perhaps we could use dictionaries with only 500 entries, if they were the *right* entries and were completely correct. We hope that future work in cross-lingual representations can answer this question.

There are several factors that may affect the cross-lingual NER F1 score resultant from these cross-lingual embeddings. We hypothesize that the most important of these are: size and quality of the monolingual corpus used to train the embeddings, and size and quality of the dictionary used to map them. In order to remove confounding variables related to shared script or character embeddings, we examine these factors in light of F1 scores from the Dictionary/nochar method in Table 18. Figure 16 shows F1 scores by Wikipedia

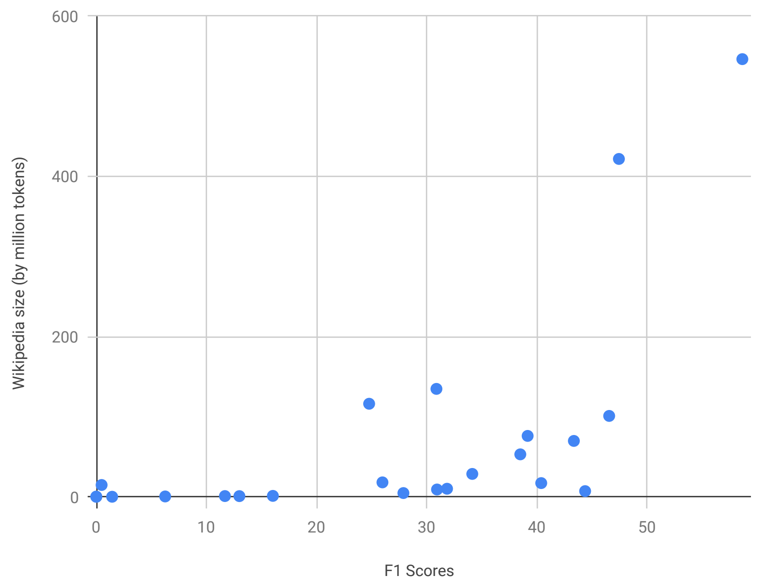


Figure 16: Scatter plot of F1 scores by size of Wikipedia. The Spearman correlation is 0.76.

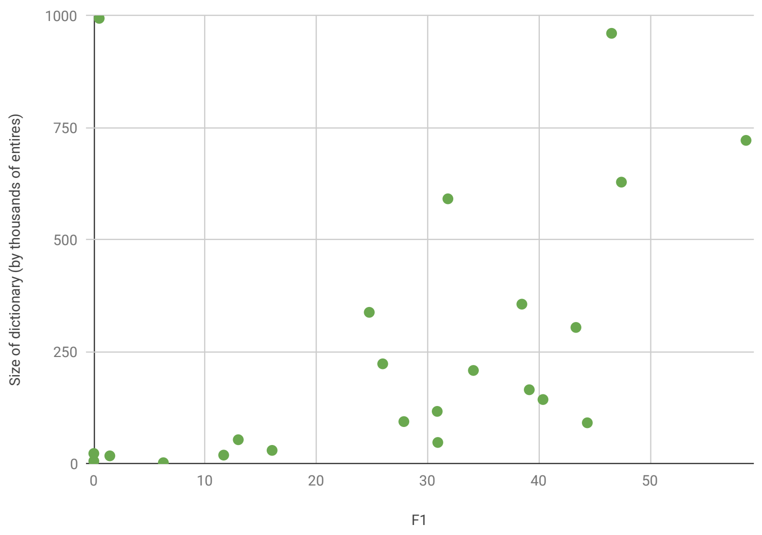


Figure 17: Scatter plot of F1 scores by size of dictionary. If we exclude the outlier in the top left corner (Chinese), the Spearman correlation is 0.83.

size, and Figure 17 shows F1 scores by dictionary size. At best we can treat the values of Wikipedia size as estimates, as we do not know the exact data used to train the monolingual fastText embeddings. Our method of size estimation is to use the Unix word count tool (wc) on a concatenated text file of each Wikipedia. We did no tokenization beforehand, so the estimates are not accurate in an absolute sense, but are relatively accurate assuming a roughly uniform distribution of punctuation across languages. We did not perform segmentation either, so the two languages that require it, Thai and Chinese, are not accurately represented.

We can also calculate the Spearman correlation for each comparison. For Figure 16, the correlation is 0.76. For Figure 17, if we ignore the outlier in the top left corner of the figure (Chinese), then the correlation is 0.83, showing a moderately strong relationship.

Final Cross-lingual Results

We now combine the best results from the cross-lingual embeddings and the other cross-lingual methods, shown in Table 19. Interestingly, the cross-lingual embeddings methods are nearly 10 points lower than the worst method, despite using the same resources. This echoes results in Xie et al. (2018) showing that the cheap translation methods consistently outperform embedding transfer.

Overall, as in the monolingual setting, mBERT embeddings are the best, but interestingly our Cheap Translation method is not far behind. It is also interesting that the difference between CT and (Xie et al., 2018) is relatively small. Across the different languages, there is no clear winner, but CT has a large improvement (greater than 5 points F1) on Akan, Chinese, and Wolof, whereas (Xie et al., 2018) has a large improvement on Hindi, Spanish, and Swahili. It's not clear what is causing this, but it may be due to embedding quality. For example, perhaps the small Wikipedia size of Akan causes particularly poor embeddings, which would lead to poor dictionary induction, therefore poor translation quality.

An interesting aspect of the mBERT embeddings is the performance on languages it has not seen. mBERT was trained on all of these languages, except for Akan, Amharic, Chinese, Somali, Wolof, and Zulu. For all of these languages (especially Amharic), the performance is never the best, but usually is non-trivial (with the exception of Amharic).

However, the cross-lingual performance of mBERT in general is surprisingly good, which echoes results from Wu and Dredze (2019). Given that no cross-lingual objective was used to train mBERT, it is baffling where this cross-lingual capability comes from. This is a fertile avenue of future research, both to understand the mechanism and to extend it.

One question we may ask, following Xie et al. (2018), is, why do the cross-lingual embedding methods produce such poor results? The Cheap Translation methods achieve 8 F1 points higher, and using mBERT results in over 10 points improvement. One answer could be related to the fact that cross-lingual embeddings are nearly always evaluated with toy bilingual dictionary induction tasks, a criticism recently levelled by Glavaš et al. (2019). When evaluated on the downsteam task of cross-lingual NER, they simply do not perform well.

Another possible answer is in the propensity of NER systems to memorize entities. **?** examined NER performance on seen and unseen entities in a test set, and showed that NER systems tend to perform far better on seen entities than unseen. In a series of small experiments, Xie et al. (2018) showed that while direct transfer using embeddings tends to work poorly (foreshadowing these results), replacing training text word embeddings with their nearest neighbors in the target space (a sort of latent translation) produced performance almost on par with cheap translation.

Finally, we look back to the monolingual results from Table 17. The average score from the best performing cross-lingual method (43.27, as shown in Table 19) is still a long way from the average score of 74.13 in the monolingual table. This suggests that there is still a lot of work to do in improving cross-lingual NER.

Language	Dict. (by script)	Cheap Translation	(Xie et al., 2018)	mBERT
aka	15.41	**42.80**	26.67	20.75
amh	11.67	**25.81**	21.72	2.88
ara	24.75	41.40	**41.54**	34.19
ben	40.37	46.12	**49.05**	44.92
cmn	0.48	**42.01**	33.89	37.62
fas	39.13	43.47	44.48	**49.94**
hin	34.11	32.99	38.95	**48.79**
hun	32.57	48.16	45.71	**61.28**
ind	53.99	57.67	58.59	**61.05**
rus	47.41	47.22	49.34	**58.69**
som	29.59	**33.79**	31.86	15.89
spa	54.48	47.04	52.51	**66.51**
swa	47.95	50.97	**56.16**	53.47
tam	25.96	32.64	32.25	**44.31**
tgl	62.71	66.78	66.58	**74.28**
tha	31.82	28.66	**33.45**	19.68
tur	48.01	51.63	53.53	**68.32**
uzb	47.30	**51.89**	50.29	51.87
vie	35.37	35.68	36.62	**55.94**
wol	15.28	**33.55**	25.51	20.29
yor	30.37	36.13	35.31	**42.67**
zul	16.49	16.92	17.03	**18.66**
avg	32.40	41.51	40.96	**43.27**

Table 19: Results from the Shared Space Experiments

5.6. Conclusions

In this chapter, we have shown experiments across a large number of languages, providing both monolingual baselines, and results with several of the most popular approaches to cross-lingual transfer. Overall, we see that cross-lingual contextual embeddings are surprisingly strong, even though they were not trained with a cross-lingual objective. We also confirm prior results that suggest that using cross-lingual embeddings in a direct transfer setting is ineffective, but that cheap translation remains a strong solution.

CHAPTER 6 : How to Build a Surprise-language NER System in One Week

6.1. Introduction

So far, this thesis has described academic work in named entity recognition (NER) aimed at cross-lingual and low-resource settings. We have made observations on realistic scenarios and we have proposed a number of techniques and have shown that they bring improved scores in certain controlled environments. But as closely as we have tried to mimic the real world, there still remains the gap between theory and practice. When it comes down to it, how does one build a low-resource NER system? In this chapter, we outline a practical protocol and best practices for developing a low-resource NER system from scratch in a situation where time is of the essence, and the identity of the language is not known beforehand ("surprise language").

To motivate this task, consider work from Microsoft researchers in 2010 after a devastating earthquake in Haiti, as described in Lewis (2010). After the earthquake hit, there was a multi-organizational effort to provide aid to those affected. One problem that aid organizations faced was a language barrier: Haitian Creole is widely spoken in Haiti, but not anywhere else. This problem was manifested in two specific ways: first in the need for English language aid manuals to be translated to Haitian Creole, and second in processing emergency hotline texts written in Haitian Creole. One solution is a Haitian Creole-English machine translation system, but no such system existed. The challenge faced by Dr. Lewis and colleagues was to develop a machine translation system with no available parallel text. Because lives were at stake, it should be developed in as short a timeframe as possible.

But at least for the aid hotline, machine translation isn't solving the problem. When deciding how to respond to these hotline texts, it is important to know **locations** of those affected, or which **organizations** are requested. By translating into English, it is now possible to run English-specific information extraction tools on the data, but it may be more efficient to develop low-resource information extraction tools directly.

We can claim experience in this scenario through participation in evaluations as part of a DARPA-funded program called LORELEI (Christianson et al., 2018), detailed in our system descriptions (Tsai et al., 2018; Mayhew et al., 2017a, 2018, 2019b). LORELEI, short for Low-REsource Languages for Emergent Incidents, is a program motivated directly by the above story of the Haitian earthquake, and has the goal of enabling "rapid, low-cost development of capabilities for low-resource languages", particularly targeting "emergent incidents" that may arise in regions with low-resource languages. These evaluations were run for 4 years (2016-2019), and I was the only team member to be involved in all evaluations, usually in a leadership role. Our results in these evaluations were always competitive with the other teams, but we achieved the top ranking NER scores in the last year of the program.

In light of our experience in this unique scenario, we have written this guide for future researchers and practitioners who find themselves in a similar situation. Because our experience is from LORELEI, the setup will largely mimic that setup. In the rest of this chapter, we will cover best practices for preparation, commonly-available resources, crunch-time development protocol, and finally comment on what results one might expect. We may summarize our recommendations as requiring time-intensive human ingenuity supplemented by cross-lingual methods from research.

6.2. Resource Assumptions

In preparation for this guide, we need to lay out the assumptions we will make about data and resources. Below we list several common resources, and the assumptions we make about availability:

Time For the sake of simplicity, and to roughly match LORELEI, we assume a timeframe of one week.

Annotated NER Data We assume no annotated data in the target language, although we can rely on annotated NER data in a high resource language. At the very minimum, this data is readily available in English (see ConLL2003 or OntoNotes), but as described in an earlier section, there are also datasets in a diverse array of languages, of which one is likely to be similar to the target.

Parallel Text Given the broad translation coverage of certain texts, such as the Bible and the Koran, it's reasonable to assume at least some parallel text. However, these are often quite small (the Bible contains roughly 30K sentences), and the domains are quite different from the target.

Monolingual Text In the world of pretraining and unsupervised learning, unannotated monolingual text is assumed to be unlimited. For many low-resource languages, however, this may not be true. While many languages have some web presence beyond Wikipedia, perhaps in local news or governmental sites, we should assume that the size of the available text is measured in the millions of tokens rather than the billions.

Human Annotators In general, we assume that speakers of the language are not easily available for annotation. In the LORELEI task, participants were provided access to native speakers of the language via conference call. These speakers, dubbed "Native Informants", or NIs for short, were neither linguists nor computer scientists, and we could assume no expertise in NLP. We were given a very limited time with them, one hour for the first checkpoint and about 4 hours for the second checkpoint, but the way we used their time was entirely up to us. Naturally, if we had unlimited access to an large number of NIs, then the problem would be analogous to any data annotation project (modulo particularities of the language and dataset). The limited availability of each NI meant that we could not count on only their annotations as the mainstay of our submissions.

Lexicons Bilingual lexicons are available in a large number of languages (Kamholz et al., 2014), although the quality and size may be variable. We can assume that a lexicon between English and the target language exists.

Romanization One obstacle to progress in this task may be the non-Latin script of some target language. Thankfully, the structure of Unicode has allowed the development of such tools as uroman (Hermjakob et al., 2018a) or unidecode[1] which convert any script to Latin script. This can often be helpful, but it is not always guaranteed to be intelligible.

6.3. Organizational Best Practices

Because this process involves trial and error and is characterized by exploration, it is crucial to define solid organizational practices at the outset. None of these recommendations depend on the target language, and can be followed if you are in a position to prepare prior to some surprise language reveal.

File formats A truism of working in NLP and data science in general is that 90% of the time is spent converting files from one format to another. In a time-sensitive situation, this can be a serious problem, so choosing a sensible file format is important. Keep in mind that it should be easy to include such information as NER annotations and romanization.

Software Directly related to the prior point is that you will need to prepare software that can read and write the file format of choice. Further, some kinds of software, especially annotation software, may need some practice to be most effective.

Folder structure We found it immensely helpful to predefine a folder structure that anticipates as much of our experiments as possible. In the heat of the moment, it can be tempting to write some results to a new folder called "best-current-anno2". While this may not be meaningful in itself, if it is written under a path called "ner/transfer/from_swahili", at least there is some context. Having this organizational context is also helpful for post-task analysis. As time allows, you will also help yourself out by keeping detailed README.txt files in each folder.

[1]`https://pypi.org/project/Unidecode/`

Shared documentation In our experience, creating a shared storage folder (we used Google Drive[2]) for documentation is important. We used shared spreadsheet software to keep track of such things as plans for submission, paths to model files, and top scores on development sets.

Physical location of team members While email and chat are useful technologies, we found that having all members of the team physically present in the same room boosted productivity significantly. This allowed us to quickly disseminate information to all others, and also to quickly resolve small problems like file permissions, or data locations.

6.4. A Protocol for Low-Resource NER

Here follows the main content of this chapter. In this section, we describe the nuts and bolts of developing a low-resource system. At this point in the timeline, the surprise language is known, and results are needed as soon as possible. We have divided it into three sections, corresponding to three stages of the process: 1) preparation, 2) exploration, 3) submission.

6.4.1. Preparation: laying the groundwork

1. Decide on a consistent tokenization. This may sound like a trivial task, but there are many subtleties to this. While many non-Latin script languages use Latin punctuation symbols, many others use their own punctuation symbols. Even building a simple sentence splitter is a non-trivial operation. You may build a high-performance system, but if the tokenization between training and target data is inconsistent, your model will fail. If your target language is one of the few that doesn't use whitespace between words, then this is an additional step you need to do.

2. Preprocess all text. Don't be afraid to throw away text, for example, if the content is too far afield, or the formatting is too odd.

3. Immediately begin training word representations. Depending on what computational

[2]`drive.google.com`

resources you have at hand, and what your algorithmic strategy is, you may want to train Brown clusters, or word vectors, or contextual embeddings. Since this can take a long time, it is crucial to start it immediately.

4. Take stock of all the resources you have at your disposal. Available resources will define strategies down the road. In particular, identify a “similar” language in which you have resources. Language similarity can be defined many ways, including according to language family, writing system, geographic similarity, and other ways. It may be useful to consult prior work on selecting a similar language (Lin et al., 2019). Is it possible to transfer these resources in a non-trivial way? For example, with IL12 (Ilocano), we found that Tagalog was so close that we could transfer models directly. On the other hand, languages such as IL11 (Odia) that have unique scripts pose different problems.

5. Learn as much about the language as possible. Read the Wikipedia page for the language, study a map of the region. Gather lists of entities that may reasonably appear in this language, including popular politicians or celebrities, local geographical landmarks such as capitals or provinces, and local organizations. Learn basic facts about the language: is it SVO or SOV? Is it influenced by another language? Does it use morphology or diacritics? Are there alternate writing systems in use? What is the history of the target region? How many speakers of this language are there? Do they also commonly speak a second language?

6. Gather a small group of documents for use as your development set. If the target domain is known, make every effort to choose a set as close as possible to the target domain. For example, if the target domain is known to be entirely made up of tweets, then choose a development set made of tweets. As you develop later parts of your system, it will be crucial to have some benchmark to guide you. Even if this development set is not fully correct with respect to annotations, we may trust that there is a relative correlation of performance between the development set and target

data. Even if this correlation proves to be small, it can be useful to have a frozen set of representative documents, at least useful for manual inspection.

7. Start annotating the development set. Even if you have access to an NI, there are multiple benefits to doing annotation yourself. First, if there are "easy" entities in the text that you can annotate, then you are saving the NI time, which they can focus on entities you can't annotate. Our research in Section 3.1 suggests that this doubles NI efficiency. Second, the practice of staring at the text will help you realize idiosyncracies of the language. For example, we noticed in Oromo (IL5), that even in news documents, there seemed to be no standard for spelling, resulting in many different forms of the same word. As another example, by examining Uyghur, we noticed that the official Latin script version of the language looks remarkably similar to Uzbek, differing only by certain consistent spelling differences. If you have access to an NI, give them the development set you have preannotated. If not, have several team members look over the development set to catch as many missing entities as possible, and once you are satisfied that you can improve it no more, freeze it for use as a benchmark. This development set can also be used for training at the end of the evaluation period.

6.4.2. Exploration: building the tools

Now that certain groundwork has been laid, the development can begin in earnest. As a general rule, all experimentation will be benchmarked against the development set created in the prior phase. Keep records of all experiments in the shared spreadsheets, including details of performance on the development set and locations of trained models or resource files. In this phase, we have identified 3 paths to success: using state-of-the-art cross-lingual techniques, building rule-based systems, and annotating data with non-speaker annotators.

Cross-lingual Techniques Using cross-lingual techniques, you can take advantage of high-resource languages, preferably languages "similar" to the target language. Try methods that you have seen proposed in research papers, but do not commit to any one. Use the embeddings you have created. Try different models, try with different parameters. Experiment with lexicons, either for cheap translation or for cross-lingual embeddings. Many of these techniques are orthogonal and can be combined. In practice, however, we have found that these techniques often give weak performance that is overshadowed by monolingual systems trained on non-speaker annotations.

All iterations of the LORELEI evaluation had a parallel track of low-resource machine translation (MT), operating on the same languages. In principle, the results from the MT track could be used as a "not-so-cheap translation" system, perhaps translating annotated text in English into the target language. We never took advantage of this, partly because of prior experience that suggests that if the translation isn't guaranteed to preserve the names, then cheap translation is better. However, it could have been a useful signal to give to the non-speaker annotators inline with the text being annotated.

One technique that we found helpful was the use of a custom-trained BERT, as outlined in our 2019 report (Mayhew et al., 2019b). The full details are in the report, but a brief description follows. We tried two different methods of training BERT: training on a few languages from scratch (which we called few-BERT, or fBERT), and continuing from a pretrained multilingual checkpoint (continue-BERT, or cBERT). When training fBERT, the training data consisted of text in English, the target language, and two or three related languages. The process for cBERT involved modifying the vocabulary of a pretrained model to include tokens in the target language, and continuing training from the pretrained checkpoint. While both methods gave high scores, the cBERT model slightly outperformed fBERT. Unfortunately, this training process is expensive both in terms of time and computational resources and may not be a feasible approach in all situations.

Rule-based Methods Gathering rules for rule-based systems is a time-honored NLP tradition. In the first year, 2016, this was one of our main techniques (Tsai et al., 2018). However, there are good reasons why the NLP community has moved on from this practice.

It is impossible to ensure that you have gathered all the right rules. In languages with heavy morphology, it can be difficult to know if a rule should match (consider Kinyarwanda which includes both prefixes and suffixes). Further, a rule may not always apply, especially if a name is also a common word.

Although we do not recommend this as a primary strategy, it is worthwhile to gather lists of popular names as a post-processing sanity check for your system. In our case, we knew that the annotation guidelines called for every twitter handle to be tagged, so we wrote rules to ensure this.

Non-Speaker Annotations In our experience, non-speaker annotation (combined with native informant annotations as available) is the most reliable path to good performance. However, the details of this process are very important.

First of all, annotation in a language you don't speak is very difficult, and not everyone can do it. We have found that qualities of a good annotator include, but are not limited to:

- Language skills, such as the ability to pick up new vocabulary and linguistic patterns.
- Attention to detail.
- Strong knowledge of the target task (especially the annotation guidelines)
- Wide geographical and political knowledge. For example, if a document mentions "Macron" and "G8", then a good annotator would know to look for many other world leaders.
- A good ear. Often one only recognizes names by sounding them out. Consider the word "pyaalestaaina", which is a romanization of a Bengali word. At first glance, this may not be recognizable, but if spoken aloud, one understands it means "Palestine."

In order to get consistent annotations, it is crucial to train all annotators in the target task. In our experience, annotators tend to assume they understand the task and skim over the guidelines, only to find that they are unequipped to handle ambiguous situations. In certain situations, some frequently occurring entity is repeatedly mislabeled and model performance is severely skewed. In the 2019 evaluation, we required all annotators to take a quiz over the annotation guidelines, and then spend 30 minutes annotating gold English text. This eliminated language difficulty as a variable, and allowed us to directly measure their performance and give feedback on common mistakes.

No matter how good the annotators, using the right tool is important. We have always used TALEN (Mayhew and Roth, 2018), although other tools exist (Lin et al., 2018).[3] One powerful feature in TALEN is the ability to easily search Google for words or phrases. We have found that these searches can bring up all sorts of useful information, including English versions of the document, or the corresponding English Wikipedia page. In some cases, the original page from which the text had been scraped is returned. We have found that the images in the document can be helpful, either by suggesting a topic (perhaps from an image of migrants in a boat), or by giving away certain recognizable landmarks (St. Peter's Basilica in Moscow), which the annotator may not have known to look for in the text.

For some languages, this task is very hard, especially at first. For example, we found Sinhalese (IL9) and Odia (IL11) to be particularly difficult because of the unique scripts. However, even for the hard languages, after you have stared at the text for a while, some patterns begin to emerge. For example, you slowly become able to identify certain stop words or verbs. Sometimes it can help to search out documents that describe international events which may include easily recognizable named entities giving invaluable clues to context.

Given that this task is time consuming, how do you know when you have enough? As with standard data scenarios, we have found that more annotations leads to higher performance, even if the quality of the annotations is suspect. However, in our experience, and as shown in Table 20, we have annotated as little as 45K tokens for a language, and as much as 144K tokens. By way of comparison, the ever popular CoNLL 2003 English dataset has about

[3]We have even known some teams to use a spreadsheet for annotation, where each row is a word, and different information (tags, transliterations, translations) is put in the columns.

Year	Language	Designator	Num Tokens Annotated
2017	Tigrinya	IL5	144,539
	Oromo	IL6	101,821
2018	Kinyarwanda	IL9	60,912
	Sinhalese	IL10	45,965
2019	Odia	IL11	57,636
	Ilocano	IL12	98,331

Table 20: Number of tokens annotated by language. This includes annotations from non-speakers (NS), and from native informants (NI). Uyghur (IL3) is missing because we did not record the number of annotations.

200K tokens in the training set.

6.4.3. Submission: preparing final results

When submission time comes, fold your development set into your training set, and train final models, perhaps with a range of parameters. Use the rules created earlier for post-processing sanity checks.

We have experimented with propagating all predicted entities throughout each test document. For example, if a model tags “Germany” once in a document, we ensure that “Germany” is tagged always in that document. However, we have seen mixed results with this. If models tend to be high precision and low recall, then this is a good strategy. On the other hand, if predictions have low precision but high recall, this should be avoided because it may propagate many incorrect annotations.

6.5. Expected Outcomes

After following the above instructions, what kind of performance should you expect? The protocol and best practices described above are by no means a silver bullet for perfect results. Any number of factors could affect the final outcome, including difficulty of the target language, number and quality of available non-speaker annotators, and amount of effort put forth. In fact, even with experience of 7 different languages, it’s hard for us to

Year	Language	Designator	Top F1
2016	Uyghur	IL3	60.4
2017	Tigrinya	IL5	75.1
	Oromo	IL6	62.5
2018	Kinyarwanda	IL9	66.8
	Sinhalese	IL10	60.5
2019	Odia	IL11	79.4
	Ilocano	IL12	79.5

Table 21: Our official evaluation results in all incident languages.

predict performance on some new language.

However, perhaps we can shed light on this by looking at what we have accomplished in the past, as shown in Table 21. Overall, this shows an average performance of 69.1 F1, although this papers over many important differences. In all cases, we had access (albeit limited) to native informants, who undoubtedly improved the quality of our annotations. In 2016, our efforts were focused on only one language, and the evaluation timeframe was a month instead of a week. However, we had no experience and we were learning on the job. In 2017, some gold annotations were provided for Tigrinya (IL5), giving it a large boost in performance. In 2019, we had our most experienced team so far, with many extra annotators (especially for IL12), and we also switched to using neural methods, including contextual embeddings.

Overall, as a rule of thumb, we believe that if one follows the procedure described above, uses strong models and embeddings, one can expect performance around 70% F1.

6.6. Conclusions

We have outlined a guide for developing an NER system for a surprise low-resource language in a compressed time frame. Our experience in 4 years of LORELEI evaluations has positioned us well to offer advice in this area. We hope that this is useful for practitioners in any scenario where a new NER system is required (emergency or not), and we also hope that the insights gathered here can spur further research into low-resource NLP.

CHAPTER 7 : Conclusion

In this thesis, we have outlined a program of research aimed at improving the state of the art in low-resource named entity recognition. Before the chapters of this thesis were first published, the best methods for cross-lingual NER required using large amounts of parallel text (Täckström et al., 2012; Wang et al., 2013a).

7.1. Summary of Contributions

In Chapter 3, we showed how one may still find certain indirect signals useful in the absence of full supervision. One fertile yet unexplored area of research is using annotations from non-speakers of a language. Prior experience has shown this to be a useful technique, but there had been little work on quantifying these capabilities. With a preliminary human experiment involving annotation of Russian, we showed that non-speakers are able to produce meaningful annotations (although still noisy) and that the quality of annotation improves even over a short time period. We showed that combining several sets of noisy annotations produces improved results, taking advantage of the fact that non-speaker annotators are far more available than native speakers. Finally, we showed that preannotating text with non-speakers can dramatically improve the efficiency of a native informant. In another project, observing that the output of non-speaker annotations is often partially annotated, we presented a method for learning from partially annotated training data, relying on constraints from prior knowledge. The chapter concluded with a study on token-internal signals of named entities, showing that names are distinguishable from non-names in many languages.

Chapter 4 moved on to cross-lingual techniques. First, we showed that a cross-lingual wikifier can produce language-independent representations. Such representations allow zero-shot transfer across languages, provided the Wikipedia in the target language is sufficiently rich. Realizing that many languages still suffer from poor Wikipedia representation, we proposed a method for "cheap translation", in which bilingual lexicons, readily available in many languages, are used to translate annotated high-resource text into some low-resource target language. We show the effectiveness of this technique with experiments on simulated low-resource languages, as well as a real-world case study in Uyghur.

The thesis closed out with two meta-projects, first Chapter 5, which evaluated several recent cross-lingual techniques over a large number of test languages, and then Chapter 6, which distilled recent experiences in building real-world low-resource NER systems to a practical guide.

7.2. What's next for low-resource NLP?

Before closing, I will discuss some areas that have promise for low-resource NLP.

Multilingual Contextualized Representations As of writing, the contextualized representation revolution is nearly 2 years old. ELMo, Flair, BERT, XLNet, and countless similarly named follow-on works have shown great promise not only in English, but also in multilingual and cross-lingual settings. In fact, the surprising success of mBERT (as shown in Chapter 5) has brought more questions that answers. If it's not trained with cross-lingual objectives, how can it work? Once we understand the mechanism, can we improve on it?

I believe that the success that we have seen with cross-lingual multisource NER (Tsygankova et al., 2019), and custom mBERTs (LORELEI evaluations) will only continue. Of particular interest are studies in which no annotated data in the target language is required.

One caveat in this area: these pretrained language models require large amounts of raw text for training, but for many languages "low-resource" may also mean that raw text is scarce, or noisy. In all future research regarding contextual embeddings for low-resource languages, it will be crucial to develop an understanding of how much text is required to achieve good results.

Human Annotations As we saw in Chapter 3, humans are surprisingly good at cross-lingual annotations, even when they don't speak the target language. I see two ways to exploit this capability, first in studying and automating the processes that humans use, and second in building expert systems that combine the best capabilities of humans and machines.

Over the course of the LORELEI evaluations, we discovered that the most surefire way to get high scores quickly was to use non-speaker annotators. In some sense, this conclusion is frustrating because the goal is to develop some smart, automatic, fully cross-lingual NLP algorithm. All the same, the process is illuminating, and it's worth asking the question: how is it that humans are so good at annotating even languages they don't understand? It was partially this question that led to the work in Section 3.3 on character language models. We noticed while annotating that one signal a human annotator used was words with relatively uncommon character sequences. Even if you don't understand what the common character sequences mean, you can detect an uncommon sequence. But we believe such observations can be taken much further. For example, humans use a good deal of reasoning during annotation, using knowledge such as, "person names often appear with titles, and usually have 1 or 2 tokens", or "location names typically remain geographically local within a document", or "entities within a document do not appear independently of each other." How to formalize and execute such reasoning is a difficult problem, not least because of the reliance on expert knowledge, such as local naming conventions, global geography, and international politics.

While studying this human knowledge will lead to fruitful research directions, on a practical level, it may always make sense to keep a human in the loop. Research in the above area will highlight which skills are particularly human, and which can be easily automated. These results can be used to build powerful interfaces for cross-lingual annotation, similar to TALEN , which offload the easy parts of annotation to algorithms and use human experts as efficiently as possible, perhaps similar to active learning (Settles, 2012).

BIBLIOGRAPHY

Ž. Agić and I. Vulić. JW300: A wide-coverage parallel corpus for low-resource languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210, Florence, Italy, July 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P19-1310`.

Ž. Agić, D. Hovy, and A. Søgaard. If all you have is a bit of the Bible: Learning POS taggers for truly low-resource languages. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 268–272, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-2044. URL `https://www.aclweb.org/anthology/P15-2044`.

Ž. Agić, A. Johannsen, B. Plank, H. Martínez Alonso, N. Schluter, and A. Søgaard. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics*, 4:301–312, 2016. doi: 10.1162/tacl_a_00100. URL `https://www.aclweb.org/anthology/Q16-1022`.

G. Aguilar, F. AlGhamdi, V. Soto, M. Diab, J. Hirschberg, and T. Solorio. Named entity recognition on code-switched data: Overview of the CALCS 2018 shared task. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 138–147, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-3219. URL `https://www.aclweb.org/anthology/W18-3219`.

R. Al-Rfou, V. Kulkarni, B. Perozzi, and S. Skiena. Polyglot-ner: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, British Columbia, Canada*. SIAM, 2015.

R. K. Amplayo, S. Lim, and S.-w. Hwang. Entity commonsense representation for neural abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 697–707, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1064. URL `https://www.aclweb.org/anthology/N18-1064`.

R. Arora, C.-T. Tsai, K. Tsereteli, P. Kambadur, and Y. Yang. A semi-Markov structured support vector machine model for high-precision named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5862–5866, Florence, Italy, July 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P19-1587`.

M. Artetxe, G. Labaka, and E. Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–

798, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1073. URL `https://www.aclweb.org/anthology/P18-1073`.

I. Augenstein, L. Derczynski, and K. Bontcheva. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech and Language*, 44:61–83, 2017.

D. Balasuriya, N. Ringland, J. Nothman, T. Murphy, and J. R. Curran. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18, Suntec, Singapore, Aug. 2009. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W09-3302`.

L. Ballesteros and B. Croft. Dictionary methods for cross-lingual information retrieval. In *International Conference on Database and Expert Systems Applications*, pages 791–801. Springer, 1996.

P. Baltescu, P. Blunsom, and H. Hoang. Oxlm: A neural language modelling framework for machine translation. *The Prague Bulletin of Mathematical Linguistics*, 102(1):81–92, october 2014. URL `https://ufal.mff.cuni.cz/pbml/102/art-baltescu-blunsom-hoang.pdf`.

L. Barrault, O. Bojar, M. R. Costa-jussà, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, P. Koehn, S. Malmasi, C. Monz, M. Müller, S. Pal, M. Post, and M. Zampieri. Findings of the 2019 conference on machine translation (wmt19). In *Proceedings of the Fourth Conference on Machine Translation*, pages 128–188, Florence, Italy, August 2019. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W19-5301`.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: 10.1162/tacl_a_00051. URL `https://www.aclweb.org/anthology/Q17-1010`.

O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, M. Huck, A. Jimeno Yepes, P. Koehn, V. Logacheva, C. Monz, M. Negri, A. Névéol, M. Neves, M. Popel, M. Post, R. Rubino, C. Scarton, L. Specia, M. Turchi, K. Verspoor, and M. Zampieri. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2301. URL `https://www.aclweb.org/anthology/W16-2301`.

P. F. Brown, V. J. D. Pietra, P. V. DeSouza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 1992.

P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2): 263–311, 1993.

J. D. Burger, J. C. Henderson, and W. T. Morgan. Statistical named entity recognizer adaptation. In *Proceedings of CoNLL-2002*, pages 163–166. Taipei, Taiwan, 2002.

X. Carreras, L. Màrques, and L. Padró. Named entity extraction using adaboost. In *Proceedings of CoNLL-2002*, 2002.

M. Chang, L. Ratinov, D. Roth, and V. Srikumar. Importance of semantic representation: Dataless classification. In *Proc. of the Conference on Artificial Intelligence (AAAI)*, 7 2008. URL `http://cogcomp.org/papers/CRRS08.pdf`.

M.-W. Chang, L. Ratinov, and D. Roth. Guiding semi-supervision with constraint-driven learning. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 280–287, Prague, Czech Republic, 6 2007. Association for Computational Linguistics. URL `http://cogcomp.org/papers/ChangRaRo07.pdf`.

O. Chapelle, B. Scholkopf, and A. Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

N. V. Chawla. Data mining for imbalanced datasets: An overview. In *The Data Mining and Knowledge Discovery Handbook*, 2005.

C. Christianson, J. Duncan, and B. Onyshkevych. Overview of the darpa lorelei program. *Machine Translation*, 32(1-2):3–9, 2018.

C. Christodouloupoulos and M. Steedman. A massively parallel corpus: the bible in 100 languages. *Language resources and evaluation*, 49(2):375–395, 2015.

K. Clark, M.-T. Luong, C. D. Manning, and Q. Le. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1217. URL `https://www.aclweb.org/anthology/D18-1217`.

A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou. Word translation without parallel data. In *Proceedings of ICLR*, 2018.

S. Cucerzan and D. Yarowsky. Language independent named entity recognition combining morphological and contextual evidence. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999. URL `https://www.aclweb.org/anthology/W99-0612`.

D. Das and S. Petrov. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, 2011.

R. Das, M. Zaheer, S. Reddy, and A. McCallum. Question answering on knowledge bases and text using universal schema and memory networks. In *ACL*, 2017.

M. Dehghani, A. Mehrjou, S. Gouws, J. Kamps, and B. Schölkopf. Fidelity-weighted learning. *CoRR*, abs/1711.02799, 2017.

L. Derczynski, S. Chester, and K. S. Bøgh. Tune your brown clustering, please. In *International Conference Recent Advances in Natural Language Processing, RANLP*, volume 2015, pages 110–117. Association for Computational Linguistics, 2015.

L. Derczynski, K. Bontcheva, and I. Roberts. Broad twitter corpus: A diverse named entity recognition resource. In *COLING*, 2016.

L. Derczynski, E. Nichols, M. van Erp, and N. Limsopatham. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, 2017.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://www.aclweb.org/anthology/N19-1423`.

V. H. Do, N. F. Chen, B. P. Lim, and M. A. Hasegawa-Johnson. Multitask learning for phone recognition of underresourced languages using mismatched transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26:501–514, 2018.

M. S. Dryer and M. Haspelmath. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig, 2013. URL `http://wals.info/`.

L. Duong, T. Cohn, K. Verspoor, S. Bird, and P. Cook. What can we get from 1000 tokens? A case study of multilingual POS tagging for resource-poor languages. In *EMNLP*. Citeseer, 2014.

G. Durrett and D. Klein. A joint model for entity analysis: Coreference, typing, and linking. *TACL*, 2014.

C. Dyer, V. Chahuneau, and N. A. Smith. A simple, fast, and effective reparameterization of ibm model 2. In *Proc. of NAACL*, 2013.

M. Ehrmann, M. Turchi, and R. Steinberger. Building a multilingual named entity-annotated corpus using annotation projection. In *RANLP*, 2011.

A. Ekbal and S. Bandyopadhyay. A web-based bengali news corpus for named entity recognition. *Language Resources and Evaluation*, 42(2):173–182, 2008.

C. Elkan and K. Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220. ACM, 2008.

J. V. Enghoff, S. Harrison, and Ž. Agić. Low-resource named entity recognition via multi-source projection: Not quite there yet? In *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 195–201, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6125. URL `https://www.aclweb.org/anthology/W18-6125`.

M. Faruqui and C. Dyer. Improving vector space word representations using multilingual correlation. In *EACL*, pages 462–471, 2014.

E. R. Fernandes and U. Brefeld. Learning from partially annotated sequences. In *ECML/PKDD*, 2011.

J. R. Finkel and C. D. Manning. Nested named entity recognition. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, 2009.

J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, 2005.

D. Flannery, Y. Miyao, G. Neubig, and S. Mori. A pointwise approach to training dependency parsers from partially annotated corpora. 2012.

R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In W. Daelemans and M. Osborne, editors, *Proceedings of CoNLL-2003*, pages 168–171. Edmonton, Canada, 2003.

J. A. Fries, S. Wu, A. Ratner, and C. Ré. Swellshark: A generative model for biomedical named entity recognition without labeled data. *CoRR*, abs/1704.06360, 2017.

E. Gabrilovich and S. Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *IJCAI*, 2007.

K. Ganchev, J. Gillenwater, and B. Taskar. Dependency grammar induction via bitext projection constraints. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 369–377. Association for Computational Linguistics, 2009.

K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 11:2001–2049, 2010.

M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters, M. Schmitz, and L. Zettlemoyer. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-2501. URL `https://www.aclweb.org/anthology/W18-2501`.

G. Glavaš, R. Litschko, S. Ruder, and I. Vulić. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 710–721, Florence, Italy, July 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P19-1070`.

J. Goldberger and E. Ben-Reuven. T raining deep neural-networks using a noise adaptation layer. 2017.

E. Grave. Weakly supervised named entity classification. In *AKBC*, 2014.

R. Grishman. Information extraction: Techniques and challenges. In *International summer school on information extraction*, pages 10–27. Springer, 1997.

N. Gupta, S. Singh, and D. Roth. Entity linking via joint encoding of types, descriptions, and context. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017. URL `http://cogcomp.org/papers/GuptaSiRo17.pdf`.

M. A. Hedderich and D. Klakow. Training a neural network in a low-resource setting on automatically annotated noisy data. 2018.

K. M. Hermann and P. Blunsom. Multilingual Models for Compositional Distributional Semantics. In *Proc. of ACL*, 2014.

U. Hermjakob, J. May, and K. Knight. Out-of-the-box universal Romanization tool uroman. In *Proceedings of ACL 2018, System Demonstrations*, pages 13–18, Melbourne, Australia, July 2018a. Association for Computational Linguistics. doi: 10.18653/v1/P18-4003. URL `https://www.aclweb.org/anthology/P18-4003`.

U. Hermjakob, J. May, M. Pust, and K. Knight. Translating a language you don't know in the Chinese room. In *Proceedings of ACL 2018, System Demonstrations*, pages 62–67, Melbourne, Australia, July 2018b. Association for Computational Linguistics. doi: 10.18653/v1/P18-4011. URL `https://www.aclweb.org/anthology/P18-4011`.

J. Hernández-González, I. Inza, and J. A. Lozano. Weak supervision and other non-standard classification problems: a taxonomy. *Pattern Recognition Letters*, 69:49–55, 2016.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9: 1735–1780, 1997.

D. Hovy and E. H. Hovy. Exploiting partial annotations with em training. 2012.

E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. Ontonotes: The 90% solution. In *Proceedings of HLT/NAACL*, 2006.

D. A. Hull and G. Grefenstette. Querying across languages: a dictionary-based approach to multilingual information retrieval. In *Proceedings of the 19th annual international ACM*

SIGIR conference on Research and development in information retrieval, pages 49–57. Citeseer, 1996.

R. Hwa, P. Resnik, A. Weinberg, C. I. Cabezas, and O. Kolak. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 2005.

T. Iwakura, K. Komiya, and R. Tachibana. Constructing a Japanese basic named entity corpus of various genres. In *Proceedings of the Sixth Named Entity Workshop*, pages 41–46, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2706. URL `https://www.aclweb.org/anthology/W16-2706`.

H. Ji, J. Nothman, and B. Hachey. Overview of tac-kbp2014 entity discovery and linking tasks. In *Text Analysis Conference (TAC2014)*, 2015.

H. Ji, J. Nothman, B. Hachey, and R. Florian. Overview of tac-kbp2015 tri-lingual entity discovery and linking. In *Text Analysis Conference (TAC2015)*, 2016.

Z. Jie, P. Xie, W. Lu, R. Ding, and L. Li. Better modeling of incomplete annotations for named entity recognition. In *Proceedings of NAACL*, 2019.

P. Jyothi and M. Hasegawa-Johnson. Transcribing continuous speech using mismatched crowdsourcing. In *INTERSPEECH*, 2015a.

P. Jyothi and M. Hasegawa-Johnson. Acquiring speech transcriptions using mismatched crowdsourcing. In *AAAI*, 2015b.

D. Kamholz, J. Pool, and S. M. Colowick. Panlex: Building a resource for panlingual lexical translation. In *LREC*, pages 3145–3150, 2014.

J. Kazama and K. Torisawa. A new perceptron algorithm for sequence labeling with non-local features. In *Proceedings of the 2007 Joint Conference of EMNLP-CoNLL*, 2007.

D. Khashabi, M. Sammons, B. Zhou, T. Redman, C. Christodoulopoulos, V. Srikumar, N. Rizzolo, L. Ratinov, G. Luo, Q. Do, C.-T. Tsai, S. Roy, S. Mayhew, Z. Feng, J. Wieting, X. Yu, Y. Song, S. Gupta, S. Upadhyay, N. Arivazhagan, Q. Ning, S. Ling, and D. Roth. CogCompNLP: Your swiss army knife for nlp. In *11th Language Resources and Evaluation Conference*, 2018a.

D. Khashabi, M. Sammons, B. Zhou, T. Redman, C. Christodoulopoulos, V. Srikumar, N. Rizzolo, L. Ratinov, G. Luo, Q. Do, C.-T. Tsai, S. Roy, S. Mayhew, Z. Feng, J. Wieting, X. Yu, Y. Song, S. Gupta, S. Upadhyay, N. Arivazhagan, Q. Ning, S. Ling, and D. Roth. Cogcompnlp: Your swiss army knife for nlp. In *LREC*, 2018b. URL `http://cogcomp.org/papers/2018_lrec_cogcompnlp.pdf`.

H. Khayrallah and P. Koehn. On the impact of various types of noise on neural machine translation. *arXiv preprint arXiv:1805.12282*, 2018.

J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182, 2003.

S. Kim, K. Toutanova, and H. Yu. Multilingual named entity recognition using parallel data and metadata from Wikipedia. In *ACL*, 2012.

D. Klein, J. Smarr, H. Nguyen, and C. Manning. Named entity recognition with character-level models. In W. Daelemans and M. Osborne, editors, *Proceedings of CoNLL-2003*, 2003.

P. Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proc. of MT Summit*, 2005.

P. Koehn and R. Knowles. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, Aug. 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-3204. URL `https://www.aclweb.org/anthology/W17-3204`.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

J. Kravalová and Z. Žabokrtský. Czech named entity corpus and SVM-based recognizer. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 194–201, Suntec, Singapore, Aug. 2009. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W09-3538`.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the International Conference on Machine Learning (ICML)*, 2001.

G. Lample and A. Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.

G. Lample, M. Ballesteros, S. K. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *HLT-NAACL*, 2016.

W. S. Lee and B. Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, 2003.

W. Lewis. Haitian creole: How to build and ship an mt engine from scratch in 4 days, 17 hours, & 30 minutes. In *EAMT 2010: Proceedings of the 14th Annual conference of the European Association for Machine Translation, Saint-Raphaël, France. 8pp.* Citeseer, 2010.

J. Li, A. Sun, J. Han, and C. Li. A survey on deep learning for named entity recognition. *arXiv preprint arXiv:1812.09449*, 2018.

Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, 2016.

Y. Lin, C. Costello, B. Zhang, D. Lu, H. Ji, J. Mayfield, and P. McNamee. Platforms for non-speakers annotating names in any language. In *Proceedings of ACL 2018, System Demonstrations*, pages 1–6, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-4001. URL `https://www.aclweb.org/anthology/P18-4001`.

Y.-H. Lin, C.-Y. Chen, J. Lee, Z. Li, Y. Zhang, M. Xia, S. Rijhwani, J. He, Z. Zhang, X. Ma, A. Anastasopoulos, P. Littell, and G. Neubig. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy, July 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P19-1301`.

X. Ling and D. Weld. Fine-grained entity recognition. In *Proceedings of the 26th Conference on Artificial Intelligence (AAAI)*, 2012a.

X. Ling and D. S. Weld. Fine-grained entity recognition. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2012b. URL `http://aiweb.cs.washington.edu/ai/pubs/ling-aaai12.pdf`.

B. Liu, W. S. Lee, P. S. Yu, and X. Li. Partially supervised classification of text documents. In *ICML*, 2002.

B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu. Building text classifiers using positive and unlabeled examples. In *ICDM*, 2003.

K. Liu, L. Xu, and J. Zhao. Opinion target extraction using word-based translation model. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1346–1356, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D12-1123`.

T. Luong, H. Pham, and C. D. Manning. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 2015.

X. Ma and E. Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1101. URL `https://www.aclweb.org/anthology/P16-1101`.

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL: System Demonstrations*, 2014.

M. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. 1993.

S. Mausam, Soderland, O. Etzioni, D. S. Weld, K. Reiter, M. Skinner, M. Sammer, J. Bilmes, et al. Panlingual lexical translation via probabilistic inference. *Artificial Intelligence*, 174(9-10):619–637, 2010.

S. Mayhew and D. Roth. TALEN: Tool for annotation of low-resource ENtities. In *Proceedings of ACL 2018, System Demonstrations*, pages 80–86, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-4014. URL `https://www.aclweb.org/anthology/P18-4014`.

S. Mayhew, C. Duncan, M. Sammons, C.-T. Tsai, X. Li, H. Pan, S. Zhou, J. Zou, and Y. Song. University of Illinois LoReHLT17 Submission. Technical report, University of Illinois, 2017a.

S. Mayhew, C.-T. Tsai, and D. Roth. Cheap translation for cross-lingual named entity recognition. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017b. URL `http://cogcomp.org/papers/MayhewTsRo17.pdf`.

S. Mayhew, S. Upadhyay, W. Yin, L. Huo, D. Jain, P. Poudyal, T. Tsygankova, Y. Chen, X. Li, N. Gupta, C. Duncan, M. Sammons, J. Sheffield, and D. Roth. University of Pennsylvania LoReHLT 2018 Submission. Technical report, University of Pennsylvania, 2018. URL `https://cogcomp.seas.upenn.edu/papers/MUYHJPTCLGDSSR18.pdf`.

S. Mayhew, S. Chaturvedhi, C.-T. Tsai, and D. Roth. Named entity recognition with partially annotated training data. In *Proc. of the Conference on Computational Natural Language Learning (CoNLL)*, 2019a.

S. Mayhew, J. Lee, F. Marini, T. Tsygankova, Z. Wang, X. Yu, X. Fu, W. Shi, Z. Zhao, W. Yin, J. Hay, K. K, M. Shur, J. Sheffield, and D. Roth. University of Pennsylvania LoReHLT 2019 Submission. Technical report, University of Pennsylvania, August 2019b. URL `no.url.yet`.

S. Mayhew, T. Tsygankova, and D. Roth. ner and pos when nothing is capitalized. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019c.

A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 188–191. Association for Computational Linguistics, 2003.

R. McDonald, S. Petrov, and K. Hall. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 62–72. Association for Computational Linguistics, 2011.

P. McNamee and J. Mayfield. Entity extraction without language-specific resources. In *Proceedings of CoNLL-2002*, pages 183–186. Taipei, Taiwan, 2002.

M. Mitrofan. Bootstrapping a Romanian corpus for medical named entity recognition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 501–509, Varna, Bulgaria, Sept. 2017. INCOMA Ltd. doi: 10.26615/978-954-452-049-6_066. URL `https://doi.org/10.26615/978-954-452-049-6_066`.

B. Mohit, N. Schneider, R. Bhowmick, K. Oflazer, and N. A. Smith. Recall-oriented learning of named entities in arabic wikipedia. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 162–173. Association for Computational Linguistics, 2012.

D. Mollá, M. van Zaanen, and D. Smith. Named entity recognition for question answering. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 51–58, Sydney, Australia, Nov. 2006. URL `https://www.aclweb.org/anthology/U06-1009`.

S. Mori, Y. Nakata, G. Neubig, and T. Sasada. Pointwise prediction and sequence-based reranking for adaptable part-of-speech tagging. In *PACLING*, 2015.

A. Moro, A. Raganato, and R. Navigli. Entity linking meets word sense disambiguation: A unified approach. In *Transactions of the Association for Computational Linguistics*, 2014.

D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

V. Nikoulina, A. Sandor, and M. Dymetman. Hybrid adaptation of named entity recognition for statistical machine translation. In *Proceedings of the Second Workshop on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid MT*, pages 1–16, Mumbai, India, Dec. 2012. The COLING 2012 Organizing Committee. URL `https://www.aclweb.org/anthology/W12-5701`.

Q. Ning, Z. Yu, C. Fan, and D. Roth. Exploiting partially annotated data for temporal relation extraction. 2018.

J. Nothman, J. R. Curran, and T. Murphy. Transforming wikipedia into named entity training data. In *Proceedings of the Australian Language Technology Workshop*, pages 124–132, 2008.

J. Nothman, T. Murphy, and J. R. Curran. Analysing Wikipedia and gold-standard corpora for NER training. In *Proceedings of the 12th Conference of the European Chapter* *of the ACL (EACL 2009)*, pages 612–620, Athens, Greece, Mar. 2009. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/E09-1070`.

J. Nothman, N. Ringland, W. Radford, T. Murphy, and J. R. Curran. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151–175, 2013.

X. Pan, B. Zhang, J. May, J. Nothman, K. Knight, and H. Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1946–1958, 2017.

H. Peng and D. Roth. Two discourse driven language models for semantics. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016. URL `http://cogcomp.org/papers/PengRo16.pdf`.

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

T. Pires, E. Schlinger, and D. Garrette. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P19-1493`.

J. Piskorski, L. Laskova, M. Marcińczuk, L. Pivovarova, P. Přibáň, J. Steinberger, and R. Yangarber. The second cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 63–74, Florence, Italy, Aug. 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W19-3709`.

B. Plank and Ž. Agić. Distant supervision from disparate sources for low-resource part-of-speech tagging. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 614–620, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1061. URL `https://www.aclweb.org/anthology/D18-1061`.

H. Poostchi, E. Zare Borzeshi, and M. Piccardi. BiLSTM-CRF for Persian named-entity recognition ArmanPersoNERCorpus: the first entity-annotated Persian dataset. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan, May 2018. European Languages Resources Association (ELRA). URL `https://www.aclweb.org/anthology/L18-1701`.

S. Pradhan, L. Ramshaw, M. Marcus, M. Palmer, R. Weischedel, and N. Xue. Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. In *CoNLL*, 2011.

S. S. Pradhan, E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. Ontonotes:

A unified relational semantic representation. *International Journal of Semantic Computing*, 2007.

B. D. Ramerth, D. R. Davidson, and J. L. Moore. Using parts-of-speech tagging and named entity recognition for spelling correction, June 28 2012. US Patent App. 12/976,849.

L. A. Ramshaw and M. P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third Annual Workshop on Very Large Corpora*, 1995.

L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *Proc. of the Conference on Computational Natural Language Learning (CoNLL)*, 6 2009. URL http://cogcomp.org/papers/RatinovRo09.pdf.

L. Ratinov and D. Roth. Glow tac-kbp 2011 entity linking system, 11 2011. URL http://cogcomp.org/papers/RatinovRo11.pdf.

A. Ratner, S. H. Bach, H. R. Ehrenberg, J. A. Fries, S. Wu, and C. Ré. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, 11 3:269–282, 2017.

R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

N. Reimers and I. Gurevych. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1035. URL https://www.aclweb.org/anthology/D17-1035.

E. Riloff et al. Automatically constructing a dictionary for information extraction tasks. In *AAAI*, volume 1, pages 2–1. Citeseer, 1993.

L. Rolston and K. Kirchhoff. Collection of bilingual data for lexicon transfer learning. Technical report, University of Washington, March 2016. URL https://www2.ee.washington.edu/techsite/papers/documents/UWEETR-2016-0001.pdf.

T. Rose, M. Stevenson, and M. Whitehead. The reuters corpus volume 1-from yesterday's news to tomorrow's language resources. In *Lrec*, volume 2, pages 827–832. Las Palmas, 2002.

D. Roth. Incidental supervision: Moving beyond supervised learning. In *Proc. of the Conference on Artificial Intelligence (AAAI)*, 2 2017. URL http://cogcomp.org/papers/Roth-AAAI17-incidental-supervision.pdf.

P. Ruch, R. Baud, and A. Geissbühler. Using lexical disambiguation and named-entity recognition to improve spelling correction in the electronic patient record. *Artificial intelligence in medicine*, 29(1-2):169–184, 2003.

Y. Samih, W. Maier, L. Kallmeyer, and H. Heine. Sawt: Sequence annotation web tool. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 65–70, 2016.

E. F. T. K. Sang. Introduction to the conll-2002 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0209010, 2002.

E. F. T. K. Sang and F. D. Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CoNLL*, 2003.

S. Sekine. Named entity: History and future, 2004.

B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

C.-W. Shih, T.-H. Tsai, S.-H. Wu, C.-C. Hsieh, and W.-L. Hsu. The construction of a Chinese named entity tagged corpus: CNEC1.0. In *Proceedings of the 16th Conference on Computational Linguistics and Speech Processing*, pages 305–313, Taipei, Taiwan, Sept. 2004. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP). URL `https://www.aclweb.org/anthology/O04-1032`.

A. Sil and A. Yates. Re-ranking for joint named-entity recognition and linking. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013.

A. K. Singh. Named entity recognition for south and south east asian languages: taking stock. In *Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages*, 2008.

J. R. Smith, C. Quirk, and K. Toutanova. Extracting parallel sentences from comparable corpora using document level alignment. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010.

R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008.

Y. Song and D. Roth. On dataless hierarchical text classification. In *Proc. of the Conference on Artificial Intelligence (AAAI)*, 7 2014. URL `http://cogcomp.org/papers/SongRo14.pdf`.

Y. Song, S. Mayhew, and D. Roth. Cross-lingual dataless classification for languages with small wikipedia presence. In *arXiv*, number arXiv:1611.04122, page 17. 11 2016. URL `http://cogcomp.org/papers/SongMR16.pdf`.

Y. Song, S. Upadhyay, H. Peng, S. Mayhew, and D. Roth. Toward any-language zero-shot topic classification of textual documents. *Artificial Intelligence*, 274:133–150, 2019.

P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii. Brat: A web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the EACL*, 2012.

A. Stolcke. Srilm-an extensible language modeling toolkit. In *INTERSPEECH*, volume 2002, page 2002, 2002.

S. Strassel and J. Tracey. Lorelei language packs: Data, tools, and resources for technology development in low resource languages. 2016.

E. Strubell, P. Verga, D. Belanger, and A. McCallum. Fast and Accurate Entity Recognition with Iterated Dilated Convolutions. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Copenhagen, Denmark, September 2017a.

E. Strubell, P. Verga, D. Belanger, and A. McCallum. Fast and accurate entity recognition with iterated dilated convolutions. In *EMNLP*, 2017b.

H. Sun, T. Bedrax-Weiss, and W. W. Cohen. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*, 2019.

C. Sutton, A. McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

G. Szarvas, R. Farkas, L. Felföldi, A. Kocsor, and J. Csirik. A highly accurate named entity corpus for hungarian. In *LREC*, pages 1957–1960, 2006.

O. Täckström, R. T. McDonald, and J. Uszkoreit. Cross-lingual word clusters for direct transfer of linguistic structure. In *NAACL*, 2012.

L. Tanabe, N. Xie, L. H. Thom, W. Matten, and W. J. Wilbur. Genetag: a tagged corpus for gene/protein named entity recognition. *BMC bioinformatics*, 6(1):S3, 2005.

E. F. Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, 2002.

E. F. Tjong Kim Sang and F. De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In W. Daelemans and M. Osborne, editors, *Proceedings of CoNLL-2003*, 2003.

C.-T. Tsai and D. Roth. Illinois cross-lingual wikifier: Grounding entities in many languages to the english wikipedia. In *Proc. of the International Conference on Computational Linguistics (COLING) Demonstrations*, 12 2016a. URL `http://cogcomp.org/papers/TsaiRo16c.pdf`.

C.-T. Tsai and D. Roth. Cross-lingual wikification using multilingual embeddings. In *Proc. of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 6 2016b. URL `http://cogcomp.org/papers/TsaiRo16b.pdf`.

C.-T. Tsai, S. Mayhew, and D. Roth. Cross-lingual named entity recognition via wikification. In *Proc. of the Conference on Computational Natural Language Learning (CoNLL)*, 2016. URL `http://cogcomp.org/papers/TsaiMaRo16.pdf`.

C.-T. Tsai, S. Mayhew, Y. Song, M. Sammons, and D. Roth. Illinois CCG LoReHLT 2016 Named Entity Recognition and Situation Frame Systems. *Machine Translation*, 2018. URL `http://cogcomp.org/papers/TMSSR18.pdf`.

Y. Tsuboi, H. Kashima, H. Oda, S. Mori, and Y. Matsumoto. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 897–904. Association for Computational Linguistics, 2008.

T. Tsygankova, S. Mayhew, and D. Roth. BSNLP2019 shared task submission: Multi-source neural NER transfer. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 75–82, 2019.

A. Ugawa, A. Tamura, T. Ninomiya, H. Takamura, and M. Okumura. Neural machine translation incorporating named entity. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3240–3250, 2018.

UNESCO. Recommendation concerning the promotion and use of multilingualism and universal access to cyberspace, 2003. URL `https://en.unesco.org/recommendation-mulilingualism`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

M. Wang and C. D. Manning. Cross-lingual projected expectation regularization for weakly supervised learning. In *TACL*, 2014.

M. Wang, W. Che, and C. D. Manning. Effective bilingual constraints for semisupervised learning of named entity recognizers. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2013a.

M. Wang, W. Che, and C. D. Manning. Joint word alignment and bilingual named entity recognition using dual decomposition. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1082, Sofia, Bulgaria, Aug. 2013b. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P13-1106`.

M. Wick, P. Kanani, and A. Pocock. Minimally-constrained multilingual embeddings via artificial code-switching. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

T. Wolfe, A. Carrell, M. Dredze, and B. Van Durme. Summarizing entities using distantly supervised information extractors. In *ProfS/KG4IR/Data: Search@ SIGIR*, pages 51–58, 2018.

F. Wu and D. S. Weld. Open information extraction using Wikipedia. In *ACL*, 2010.

S. Wu and M. Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. *arXiv preprint arXiv:1904.09077*, 2019.

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

J. Xie, Z. Yang, G. Neubig, N. A. Smith, and J. Carbonell. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1034. URL `https://www.aclweb.org/anthology/D18-1034`.

V. Yadav and S. Bethard. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158, 2018.

X. Yao, B. Van Durme, and P. Clark. Automatic coupling of answer extraction and information retrieval. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 159–165, Sofia, Bulgaria, Aug. 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P13-2029`.

D. Yarowsky and G. Ngai. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proc. of NAACL*, 2001.

D. Yarowsky, G. Ngai, and R. Wicentowski. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, 2001.

X. Yu, S. Mayhew, M. Sammons, and D. Roth. On the strength of character language models for multilingual named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3073–3077, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1345. URL `https://www.aclweb.org/anthology/D18-1345`.

D. Zelenko, C. Aone, and A. Richardella. Kernel methods for relation extraction. *Journal of machine learning research*, 3(Feb):1083–1106, 2003.

D. Zeman and P. Resnik. Cross-language parser adaptation between related languages. In *IJCNLP*, 2008.

B. Zhang, X. Pan, T. Wang, A. Vaswani, H. Ji, K. Knight, and D. Marcu. Name tagging for low-resource incident languages based on expectation-driven learning. In *NAACL*, 2016.

B. Zoph, D. Yuret, J. May, and K. Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, Nov. 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1163. URL `https://www.aclweb.org/anthology/D16-1163`.