Improving NER Tagging Performance in Low-Resource Languages via Multilingual Learning

RUDRA MURTHY, Indian Institute of Technology Bombay, India
MITESH M KHAPRA, Indian Institute of Technology Madras, India
PUSHPAK BHATTACHARYYA, Indian Institute of Technology Bombay, India

Existing supervised solutions for Named Entity Recognition (NER) typically rely on a large annotated corpus. Collecting large amounts of NER annotated corpus is time-consuming and requires considerable human effort. However, collecting small amounts of annotated corpus for any language is feasible, but the performance degrades due to data sparsity. We address the data sparsity by borrowing features from the data of a closely-related language. We use hierarchical neural networks to train a supervised NER system. The feature borrowing from a closely-related language happens via the shared layers of the network. The neural network is trained on the combined dataset of the low-resource language and a closely-related language, also termed as Multilingual Learning. Unlike existing systems, we share all layers of the network between the two languages. We apply multilingual learning for NER in Indian languages and empirically show the benefits over monolingual deep learning system and a traditional machine learning system with some feature engineering. Using multilingual learning, we show that the low-resource language NER performance increases mainly due to (a) increased named entity vocabulary (b) cross-lingual sub-word features and (c) multilingual learning playing the role of regularization.

CCS Concepts: • **Computing methodologies → Information extraction**;

Additional Key Words and Phrases: Named Entity Recognition, Deep Learning, Multilingual Learning, Low-Resource Languages, Indian Languages

ACM Reference Format:
Rudra Murthy, Mitesh M Khapra, and Pushpak Bhattacharyya. 2018. Improving NER Tagging Performance in Low-Resource Languages via Multilingual Learning. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 9, 4, Article 39 (August 2018), 20 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

Named Entity Recognition (NER) plays a crucial role in several Natural Language Processing (NLP) tasks such as Information Extraction, Question Answering, *etc.* As many languages have limited training data (*data sparsity*), earlier works on NER focused on feature engineering using several machine learning algorithms [30, 33]. These systems typically use POS features, morphological information, gazetteer features and other handcrafted features to address data sparsity. *Low Resource Languages* however, lack tools like morphological analyzer, POS tagger, *etc.* or the knowledge

Authors' addresses: Rudra Murthy, Indian Institute of Technology Bombay, Powai, Mumbai, Maharashtra, 400076, India, rudra@cse.iitb.ac.in; Mitesh M Khapra, Indian Institute of Technology Madras, Chennai, Tamil Nadu, 600036, India, khapra.mitesh@gmail.com; Pushpak Bhattacharyya, Indian Institute of Technology Bombay, Powai, Mumbai, Maharashtra, 400076, India, pb@cse.iitb.ac.in.

2375-4699/2018/8-ART39 $15.00
https://doi.org/0000001.0000001

resources required for feature engineering. This hinders the successful application of existing feature-engineered systems for NER task in these languages.

Use of cross-lingual resources [3, 9, 44] as a way to mitigate data sparsity has been explored in the literature. Recently, multilingual learning using deep neural networks has been shown as a way to tackle data sparsity in low-resource languages [17]. *Multilingual Learning* (MLL) can be seen as an instance of multi-task learning where the deep neural network is trained for the same task in multiple languages by sharing some or all layers of the neural network between related languages. The model learns cross-lingual features via the shared layers (or weights) of the network. This enables the model to generalize better for the task in the primary language. Multilingual learning using deep neural networks has the advantage of not requiring parallel corpus and learns better cross-lingual features from the assisting language training data.

Multilingual learning works best when the languages are closely-related. This makes multilingual learning ideal for NER task in Indian languages. Dravidian and Indo-Aryan languages in the Indian sub-continent share the same set of phonemes and the correspondence between characters across scripts can be easily established [39]. This allows for sharing of sub-word information across languages for NER task. Indian languages have high lexical overlap [20] and similar word order [39], which allows for sharing of syntactic and semantic information across languages.

To this end, we use multilingual learning to improve the NER performance in a low-resource language (primary language) by sharing features with an assisting language. The neural network is trained on the combined data of both the primary and the assisting languages by sharing all layers of the network across both the languages. We first show the benefits of multilingual learning on German NER using either English, Spanish, and Dutch as assisting languages. We also show benefits for Spanish and Dutch NER with English as the assisting language. For low-resource language setup, we consider Marathi, Bengali, Tamil and Malayalam as the primary languages and Hindi as the assisting language in our experiments. We use bilingual embeddings to facilitate greater transfer of semantic information across languages. Bilingual embeddings have the advantage that similar words across languages lie closer in the embedding space. For Indian languages, we take advantage of the correspondence between the characters across different Indic scripts and map them to a common vocabulary. Specifically, we convert the scripts of Bengali, Tamil and Malayalam to Devanagari script. This allows for sharing of sub-word features across Indian languages for NER task.

Following are the contributions of the paper:

- We explore completely shared architectures for multilingual NER
- To the best of our knowledge, ours is the first work on multilingual NER using deep learning for low-resource Indian languages
- We empirically show the benefits of multilingual NER over traditional machine learning methods with limited feature engineering
- We present a detailed analysis on the benefits of multilingual learning

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes the proposed multilingual learning for NER. Section 4 describes the datasets and the network configurations used. Section 5 discusses results of our proposed multilingual learning, the improvements over the corresponding monolingual setting, and error analysis. Section 6 concludes the work and discusses possible future directions.

2 RELATED WORK

Our work is related to the work on using multilingual learning for deep learning based NER. We summarize the related work in this section.

Systems	Word Level		Sub-word Level	
	CNN	Bi-LSTM	CNN	Bi-LSTM
Hammerton [14]		✓		
Collobert et al. [5]	✓			
dos Santos et al. [7]	✓		✓	
Huang et al. [15]		✓		
Chiu and Nichols [4]		✓	✓	
Murthy and Bhattacharyya [28]		✓	✓	
Lample et al. [23]		✓		✓
Yang et al. [44]		✓		✓
Ma and Hovy [27]		✓	✓	
Rei [31]		✓		✓
Peters et al. [29]		✓		✓
Liu et al. [26]		✓		✓

Table 1. Overview of deep learning network design/configurations for NER

2.1 Deep Learning for NER

Table 1 shows a quick overview of existing deep learning solutions for NER and the comparisons of their architectures. The table also mentions if the systems use Bi-LSTMs or CNNs to extract word and sub-word features. Neural networks were first explored in the context of named entity recognition by Hammerton [14] but, Collobert et al. [5] were the first to successfully use neural networks for several NLP tasks including NER. Unlike existing supervised systems at that time, they used minimal handcrafted features and relied on automatically learning word representations from large unannotated corpora. The output layer was a CRF layer which modeled the entire sequence likelihood. They also used the idea of sharing network parameters across different tasks (but not between different languages).

This idea was further developed by Dos Santos and Zadrozny [8] and dos Santos et al. [7] to include sub-word features in addition to word level information. They used Convolutional Neural Networks (CNNs) with fixed filter width to extract relevant sub-word features. The combined sub-word features and word embeddings were fed to a time delay neural network as in Collobert et al. [5] and used for Spanish and Portuguese NER.

Later works on NER preferred use of Bidirectional Long Short Term Memory (Bi-LSTMs) [32] for encoding word sequence information for sequence tagging. For examples Huang et al.[15] use LSTMs for encoding word sequences and then use CRFs for decoding tag sequences. Chiu and Nichols [4] use a combination of Bi-LSTMs at the word-level with character-level CNNs for NER. The decoder is still a CRF which was trained to maximize the entire sequence likelihood. Both these approaches also use some handcrafted features. The system by Murthy and Bhattacharyya [28] was similar to the earlier porposed approaches with the exception of using softmax layer at the output. Very recently, Lample et al. [23] proposed Hierarchical Bi-LSTMs as an alternative to CNN-Bi-LSTMs wherein they first use a character level Bi-LSTMs followed by a word level Bi-LSTMs, thus forming a hierarchy of LSTMs. They also used CRF at the output layer. The model was tested on English, Spanish, Dutch, and German languages. They reported state-of-the-art results when systems with no handcrafted feature engineering are considered.

We observe that several deep learning systems with similar architecture have been proposed for NER task. Most of these systems obtain similar NER tagging performance on standard CoNLL 2003

English NER dataset[37]. Additionally, these systems report results using different word embeddings making the comparison unfair. However, the multilingual learning approach proposed in this paper is architecture independent, and, can be used with any of the above-mentioned systems.

Very recently, Rei [31], Liu et al. [26] and Peters et al. [29] used semi-supervised learning to improve the performance for sequence labeling tasks. The models jointly optimize a language modeling objective and a sequence labeling objective during training. It will be an interesting exercise to compare our multilingual learning approach with the above mentioned semi-supervised learning approaches.

2.2 Multilingual Learning using Traditional Machine Learning Models

Earlier NER systems were largely benefited by the use of word clusters [38]. These word clusters were predominantly obtained from monolingual data. Faruqui [9] explored sharing of word clusters across languages to improve NER performance. Two approaches were proposed to share word clusters across languages. In the first approach, the word clusters trained from multiple languages were used as is. Given a word, all the clusters in which the word appears were identified which were later added as corresponding cluster features for the word. In the second approach, the author proposed an heuristic to merge words appearing only in the assisting language, thereby improving the recall of the system.

Multilingual guidance for NER has also been extensively studied [25, 41–43]. The techniques rely on the availability of parallel corpus to improve NER tagging. Li et al. [25] rely on the availability of tagged parallel corpus between languages. They use contextual clues coming from the other language to improve the NER performance in both the languages. NER for parallel bi-texts has also been extensively studied [41–43]. They improve monolingual taggers by enforcing named entity tag agreement between the two languages. Some approaches require word alignment to be explicitly specified whereas other approaches jointly address word alignment and bilingual NER objective. The requirement of parallel corpus makes it infeasible to apply these models on low-resource languages. Multilingual learning using deep neural networks do not need parallel corpus, which sets it apart from these approaches for NER task.

2.3 Multilingual Learning in NLP via Deep Learning

Our work on multilingual NER is motivated by the recent works on multilingual neural machine translation [11, 17]. The core idea is to share features between languages by sharing some or all layers of the neural network. While some approaches have combination of language-specific layers and language independent layers [11], other approaches share all of the network components across languages [17]. Multilingual NER using deep learning has been explored to a certain extent in the literature. Gillick et al. [12] proposed a novel encoder-decoder architecture for NER. The input to the model is a sequence of bytes, and outputs spans with the corresponding labels. As the model works at byte-level, it was extended to the multilingual setting and substantial improvements were observed. A major drawback is that the model fails to beat other deep learning systems trained using word embeddings [27, 28, 44]. It is feasible to obtain word embeddings for many languages as they are learned on unannotated corpus which can be obtained with relative ease. Yang et al. [44] extended Lample et al. [23] approach for multilingual NER on European languages by sharing sub-word features between the language pairs. The approach reported improvements for Spanish and Dutch languages using multilingual learning. The multilingual learning approaches using deep learning do not require parallel corpus and use training data available from the languages involved.

We address the data sparsity in low-resource languages via multilingual learning using deep neural networks. Our work builds on the work of Yang et al. [44] for multilingual NER. Their approach was limited to sharing of sub-word features between languages. Availability of tools to create bilingual word embeddings [35] using cheap bilingual dictionary makes it possible to share word features across languages. It also makes sense to share all layers of the network for Indian languages, as they exhibit high lexical overlap [20], similar word order, grammar and character-level morphologies. We map the scripts from different Indian languages to Devanagari script to facilitate sharing of sub-word features across languages.

3 MULTILINGUAL LEARNING FOR NER

Low-resource languages (primary languages) have very limited NER training data. This results in many unseen named entities tagged as non-named entities during testing. *However, some of these unseen named entities or their orthographic variations might appear in the training data of another language.* In this paper, we train a hierarchical neural network for NER task by sharing all layers of the network between the primary and the assisting languages. The neural network learns cross-lingual features from the combined data of the involved languages, leading to improvements in the NER performance on the primary language. We now formalize the multilingual learning task and later describe our proposed solution.

3.1 Task Definition

Consider the training data, D_P and D_A from the primary and assisting languages respectively. Here, the training data D_P consists of word-tag sequence pairs *i.e,* $D_P = (X^i, Y^i)_{i=1}^n$, where X^i is the i^{th} sentence from the primary language and Y^i is the corresponding tag sequence, n is the number of such training sentences. Similarly, D_A consists of word-tag sequence pairs from the assisting language. The goal of multilingual learning is to find parameters θ, which maximizes the log-likelihood of the training data as given in the equation 1.

$$\underset{\theta}{\text{maximize}} \quad \log(P(D_P)) + \beta \log(P(D_A)) \tag{1}$$

Here, the hyper-parameter β, is the weight assigned to the objective of the assisting language. Assisting language being resource-rich has comparatively larger number of training instances than the primary language. The value of β controls the importance given to the assisting language objective. Setting β to zero gives the standard monolingual objective of NER.

3.2 Network Architecture

The multilingual learning strategy can be used with any of the existing deep learning systems for NER presented in the Related Works section. We now present the deep learning model proposed by Murthy and Bhattacharyya [28]. The architecture of the model is as shown in Figure 1. Given the sequence of words in a sentence as input, the model first extracts word embedding features for every word in the sentence. The model employs word lookup table to extract word embedding features. The model also employs Convolution Neural Networks (CNN) to extract sub-word features. The goal of this layer is to extract relevant sub-word features like *capitalization feature, affix features, etc.* The input to this layer is the sequence of characters forming the word. The output from various CNN filters are combined with word embeddings to obtain the final word representation.

The Figure 2 shows a sample illustration of CNN layer of width 3. For example, given the word *Lohagad*, beginning and end markers are appended, and every character is represented using one-hot encoding. The convolution filter looks at very trigram character sequence and extracts a set of k features. Max-pooling is applied to find the most relevant set of features *i.e,* max-pooling is applied every feature to find the trigram character sequence contributing for that particular feature. The CNN layer acts as feature extractors *i.e,* it helps in finding the character ngram sequence most

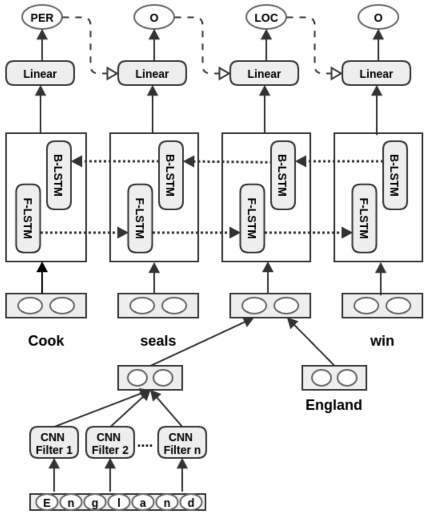


Fig. 1. Deep learning system for NER used by Murthy and Bhattacharyya 2016

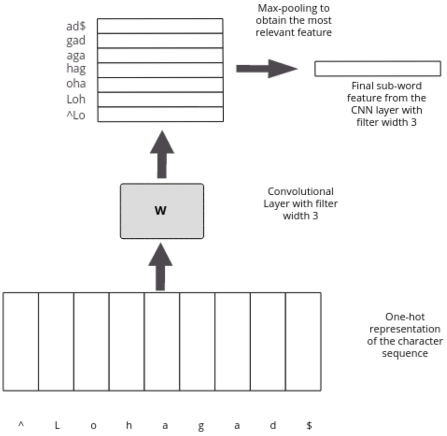

ad$
gad
aga
hag
oha
Loh
^Lo
Max-pooling to obtain the most relevant feature
Final sub-word feature from the CNN layer with filter width 3
W
Convolutional Layer with filter width 3
One-hot representation of the character sequence
^ L o h a g a d $

Fig. 2. Convolutional Neural Network extracting trigram features for the word *Lohagad*

relevant for making the NER prediction, in the above example the major feature contributor should be the character sequence *gad*.

Additionally, handcrafted features can be added to the deep learning model. For example, to add POS features we send the one-hot representation of the POS tag through a POS lookup table and obtain POS embeddings. The POS embeddings along with word embeddings and sub-word representation are concatenated to form the final word representation.

The sequence of obtained word representations for every word in the sentence is fed as input to the *Disambiguation Layer*. The *Disambiguation Layer* employs Bi-LSTM layer to disambiguate the word, where the disambiguation is with respect to the named entity task. The Bi-LSTM layer employs forward and backward LSTMs which runs over the sequence of word representations. The obtained representations from forward and backward LSTMs are concatenated to get an instance-level or sentence-level representation for the word.

The work by Murthy and Bhattacharyya [28] employ a linear layer followed by softmax function as output layer. The output from Bi-LSTM layer for every word is concatenated with the correct tag of the previous word and is fed as input to the *Output Layer*. The linear layer assigns a score for all possible tags for the current word and the scores are now sent to the softmax layer. The softmax layer converts the predicted scores to a probability score. The model is trained to minimize the negative log-likelihood of the data. As correct previous tags are unknown during testing, the model employs beam search to find the best tag sequence.

3.3 Training Strategies

The deep learning model for NER in general consists of four main components as shown in Figure 3. The *Word Feature Extractor* consists of the lookup table which are initialized with pre-trained word embeddings. The *Sub-word Feature Extractor* consists of the CNN layer which takes in character sequence as the input and outputs sub-word representation. The Bi-LSTM layer forms the *Disambiguation* layer and finally the *Output* layer. In this paper, we experiment with following training strategies where some or all components of the neural network are shared across languages.

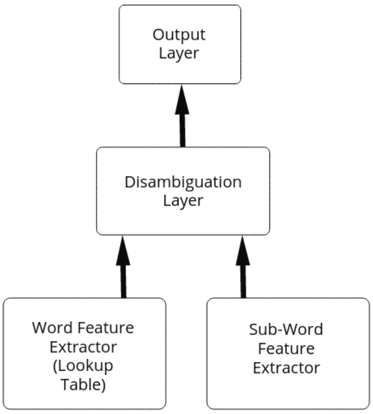

Output Layer
Disambiguation Layer
Word Feature Extractor (Lookup Table)
Sub-Word Feature Extractor

Fig. 3. Components of the Deep Learning Systems for NER

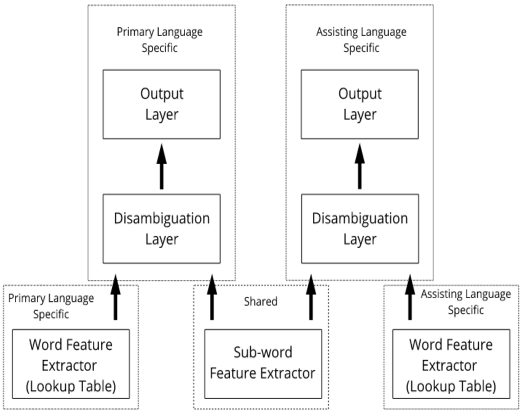

Primary Language Specific
Output Layer
Disambiguation Layer
Assisting Language Specific
Output Layer
Disambiguation Layer
Primary Language Specific
Word Feature Extractor (Lookup Table)
Shared
Sub-word Feature Extractor
Assisting Language Specific
Word Feature Extractor (Lookup Table)

Fig. 4. Network architecture for multilingual joint training via sharing sub-word features across languages

Monolingual Training

The model is trained only on the training data of the primary language. We refer to this configuration as *CNN Bi-LSTM*.

Shared Sub-Word Features

In this configuration, we share only the *Sub-word Feature Extractor* across languages (Figure 4). The primary language sentence passes through the primary language specific word feature extractor and the shared sub-word feature extractor. The final word representation is sent through the primary language disambiguation layer followed by the primary language specific output layer. Similarly, the assisting language sentence passes through the assisting language specific word feature extractor and the shared sub-word feature extractor. The word representation obtained now is sent through the assisting language specific disambiguation and output layers. This strategy of sharing only the sub-word features was proposed by Yang et al. [44]. We refer to this model as *CNN Bi-LSTM Sub-word*.

During training, the sub-word feature extractor gets updated by the error from both the primary and the assisting language disambiguation layer. The sub-word feature extractor should now detect both the primary language specific sub-word features as well as the assisting language specific sub-word features. The sharing of sub-word features is useful only if the sub-word features in the assisting language are also encountered in the primary language. As the higher layers of the network (disambiguation and output layer) are language-specific, any sub-word pattern from the assisting language may be handled differently by the primary language specific layers, unless the sub-word patterns were already encountered in the primary language training data.

Multilingual Learning

This approach is same as the *Monolingual Training* except that the model is trained on the combined training data from both the languages in a language independent way. We use bilingual word embeddings obtained from any of the existing approaches for creating bilingual word embeddings. Bilingual embeddings allows synonymous words across languages have similar embeddings features, allowing the model to share word embedding features across languages. We refer to this model as *CNN Bi-LSTM All*.

CNN Bi-LSTM All model does not suffer the same limitations as *CNN Bi-LSTM Sub-word* and can learn better cross-lingual features. Unlike *CNN Bi-LSTM Sub-word*, any sub-word pattern need not appear in both the language training data. The sub-word pattern from the assisting language also influences the decision when it appears on the primary language during test time as the higher *disambiguation* and *output* layers are also shared across languages. This, however, might also be a weakness for the *CNN Bi-LSTM All* model as the model is prone to entity drift coming from the assisting language.

4 EXPERIMENTAL SETUP

In this section, we describe the languages and the corresponding datasets used in our experiments.

4.1 Languages

We run our experiments on European languages (German, Spanish, Dutch, and Czech) and later apply the model on low-resource Indian languages. We choose German and Czech as the primary languages with English or Spanish or Dutch as assisting languages. Additionally, we also run experiments with Spanish or Dutch as primary languages and English as the assisting language. German CoNLL NER data required a special license, instead we used the publicly available German NER data released by Faruqui and Padó [10]. This data was constructed by manually annotating the *first two German Europarl session transcripts* with NER labels following the CoNLL 2003 annotation guidelines. We use the first session to create train and valid splits. Note that the German NER data is in IO format, so only for experiments involving German we convert the data in other languages also to IO format. The remaining European languages followed the IOBES annotation format. Czech NER data had different tagset compared to English or Spanish or Dutch. Hence, for the *CNN Bi-LSTM All* configuration, we do not share the output layer across languages. The datasets and their statistics are presented in Table 2.

We run our experiments on several Indian languages *viz* Hindi, Bengali, Marathi, Tamil and Malayalam. Except for Hindi, the remaining languages have less training data and hence, can be considered as low-resource languages. We obtain Hindi, Bengali, Tamil and Malayalam data from Lalitha Devi et al. [22]. We use in-house NER annotated data for Marathi. For all of these languages, we consider only the standard CoNLL tags *i.e,* Person, Location and Organization. We use 70% of the data (sentences) for training, 20% as test split and remaining as development split. IOB notation is used in all our experiments for Indian languages.

All the Indian languages, except Bengali has POS information in the provided dataset. The CNN Bi-LSTM model can also be used to train a POS tagger. We train a Bengali POS tagger using POS annotated data obtained from ILCI Bengali Tourism Corpus [16] using CNN Bi-LSTM model. The trained Bengali POS tagger is used to obtain POS tags for the Bengali NER annotated corpus.

Named entities in Indian languages also appear as non-named entities. For example, the word चन्द्र (*chandra*) in Hindi could be name of a person or could refer to the moon. Additionally, these languages do not exhibit capitalization feature as in the case of English. These factors make NER task challenging for Indian languages. Table 3 shows the percentage of named entities which appear as non-named entities in the training data[1] . It is observed that among the named entities, *Organization* entities are often most ambiguous.

[1] The statistics in the table is incomplete, as most of the languages are morphologically rich. We employ exact word match to get the overlapping entities count and may miss the inflected forms of the entities

Language	#Train Tokens	#Test Tokens	Train-Test Overlap (%)	Reference
English	204567	46666	51.39	Tjong Kim Sang and De Meulder [37]
Spanish	264715	51533	62.73	Tjong Kim Sang [36]
Dutch	202931	68994	39.61	Tjong Kim Sang [36]
German	74907	20696	42.71	Faruqui [9]
Czech	159542	20053	48.92	Konkol and Konopík [19]
Hindi	81814	23695	47.20	Lalitha Devi et al. [22]
Bengali	34386	7613	39.32	Lalitha Devi et al. [22]
Tamil	66136	18643	40.71	Lalitha Devi et al. [22]
Malayalam	26292	8274	41.89	Lalitha Devi et al. [22]
Marathi	71296	36580	16.07	In-house

Table 2. Dataset Statistics

	Hindi	Bengali	Tamil	Malayalam	Marathi
Person	35.58	24.41	19.85	12.74	17.48
Location	33.51	18.85	21.09	13.09	22.57
Organization	60.65	33.55	43.31	26.75	35.13

Table 3. Percentage of Named Entities appearing as non-named entity words in the training data

For Indian languages, we try to improve the NER tagging performance for Bengali, Marathi, Malayalam and Tamil by considering Hindi as the assisting language. We convert the annotated data of Bengali, Tamil and Malayalam to Devanagari script to facilitate better sharing of sub-word features using Indic-NLP-Library tool [21] [2] . The choice of Devanagari script is arbitrary as the goal is to have a common character script for the involved languages.

4.2 Network Details

The deep learning system is trained using Stochastic Gradient Descent method with a batch size of 1. The maximum number of steps for Backpropagation Through Time for the Bi-LSTM layer is set to 10. We alternatively pick an example from the primary and assisting language and train the model in the multilingual setting. The value of β is set to 0.1 indicating less importance given to the assisting language data.

For experiments involving German, Spanish, Dutch, and Czech, we do not perform over-sampling and the β value is set to 0.1. We use CNN filters of width 1 to 9 where CNN looks at ngram characters with n varying from 1 to 9, extracting 15 set of features each. The intuition for choosing CNN filter of width 1 is that this filter will be able to capture capitalization feature. The word embeddings obtained are of 200 dimensions. For monolingual training, the Bi-LSTM hidden layer neurons are set to 200 dimensions and for multilingual learning, they are set to 300 dimensions. We use a initial learning rate of 0.4 .

In all our Indian language experiments, we use CNN filters of width 2 to 6 where CNN looks at ngram characters with n varying from 2 to 6, extracting 50 set of features each. All pre-trained word embeddings are of dimension 300. For monolingual training, the Bi-LSTM hidden layer has 400 neurons, whereas for bilingual training it is set to 500 neurons. A learning rate of 0.04 is used in all our experiments. We constantly monitor the error on validation data. If we observe an increase

[2] https://github.com/anoopkunchukuttan/indic_nlp_library

in the error on the validation data, we load the saved previous iteration model and multiply the learning rate by 0.7. The training is stopped once the learning rate becomes less than 0.002. The Dropout layer is added before the Bi-LSTM layer for regularization. For multilingual learning, we oversample the primary language sentences to match the number of sentences in the assisting language training data.

4.3 Word Embeddings

We use Bilbowa algorithm [13] with default settings to train the bilingual word embeddings. Bilbowa takes both monolingual and bilingual corpora as input. For bilingual corpora, we use the relevant source-target portion of Europarl corpus [18] and *Opus* [34]. For monolingiual corpora, we use short abstracts for each of the 4 languages from Dbpedia [24]. The word embedding dimensions were set to 100 (default value) for our experiments on European languages.

We choose the pre-trained word embeddings from Bojanowski et al. [2] for Indian languages in our experiments. The word embeddings are of 300 dimensions and monolingual in nature *i.e,* trained on the monolingual unannotated corpus. We use the tool by Smith et al. [35] to induce bilingual word embeddings. For a language pair, the bilingual dictionary is obtained from IndoWordnet [1] to induce the bilingual word embeddings.

4.4 Baseline Systems

We consider 3 approaches as our baseline systems. The *CNN Bi-LSTM* model trained on the primary language data and the *CNN Bi-LSTM Sub-word* model [44] are the deep learning baselines. We wish to compare the results from the deep learning system with traditional machine learning approaches. We choose CRF as baseline system and employ standard set of features usually employed in traditional machine learning approach [30]. It has been established in the literature that CRF obtains better NER tagging performance when used as output layer of a deep learning system. The goal here is not to compare CRF with deep learning system, but a comparison of deep learning systems with traditional machine learning approaches using feature engineering for NER. Deep learning systems with CRF at the output layer is complementary to the deep learning architecture chosen in this paper. The multilingual learning approach can be applied to the deep learning systems employing CRF layer, in which case, the CRF layer would be shared across languages in the *CNN Bi-LSTM All* configuration.

For traditional machine learning approach, we choose features which can be readily obtained from the training corpus and do not require any language expertise. We employ the following set of features for the CRF system: unigram features, context words within a context window of 3, ngram character prefixes and affixes with *n* ranging from 2 to 4 and POS features. Here, we wish to compare a non-deep learning system using minimal feature engineering with a deep learning system.

5 RESULTS

The following section describes the results from our experiments.

5.1 Simulated Resource-Constrained Results

We now discuss our results in the simulated resource constrained setup. In our primary experiments, we treat German, Czech as the primary languages with English, Spanish, and Dutch as the assisting languages. The reason for choosing German or Czech as primary languages is that the NER data available is relatively small as compared to the English, Spanish, and Dutch datasets (thus naturally forming a pair of resource-rich (English, Spanish, Dutch) and resource poor (German, Czech) languages). We train our model jointly using the entire assisting (English or Spanish or Dutch) and the primary (German, Czech) language data. We report separate results for the case when (i) the convolutional filters are shared *(CNN Bi-LSTM Sub-word)* (ii) all layers are shared *(CNN Bi-LSTM All)*. We compare these results with the case when we train a model using only the primary language data. The results are summarized in Tables 4 and 5 with *Sub-word* and *All* referring to *CNN Bi-LSTM Sub-word* and *CNN Bi-LSTM All* configurations respectively.

Training Data	Shared	F1 Score
German	-	84.86
German + English	All	**88.18**
German + English	Sub-word	86.07
German	-	82.57
German + Dutch	All	**87.78**
German + Dutch	Sub-word	83.97
German	-	77.12
German + Spanish	All	**85.74**
German + Spanish	Sub-word	84.06

Table 4. Comparison of various multilingual learning strategies and monolingual deep learning system on German NER using English, Spanish and Dutch as assisting languages

Training Data	Shared	F1 Score
Czech	-	**80.12**
Czech + English	All	76.31
Czech + English	Sub-word	78.17
Czech	-	**78.11**
Czech + Dutch	All	76.38
Czech + Dutch	Sub-word	77.69
Czech	-	**79.40**
Czech + Spanish	All	77.24
Czech + Spanish	Sub-word	78.52

Table 5. Comparison of various multilingual learning strategies and monolingual deep learning system on Czech NER using English, Spanish and Dutch as assisting languages

Training Data	Shared	F1 Score
Spanish	-	81.16
Spanish + English	All	**82.65**
Spanish + English	Sub-word	82.38
Dutch	-	81.51
Dutch + English	All	**82.92**
Dutch + English	Sub-word	81.77

Table 6. Comparison of various multilingual learning strategies and monolingual deep learning system on Spanish and Dutch NER using English as the assisting language

We observe larger benefits from sharing all layers for German NER compared to sharing only sub-word feature extractors. However, this is not the case for Czech NER. Due to language divergence between Czech and English, Spanish, Dutch, we do not observe any benefits from multilingual learning. We observe a drop in Czech NER performance from multilingual learning. The drop, however is significant when all layers are shared compared to sharing only sub-word features. When only sub-word feature extractors are shared, the higher layers are language-dependent. The shared sub-word feature extractor is responsible for learning cross-lingual sub-word features and the model is relatively less immune to language divergence as higher layers can learn language-specific features. However, this is not the case when all layers are shared across languages.

Approach	**Tamil**	**Malayalam**	**Bengali**	**Marathi**
CRF + POS	44.60	48.70	52.44	44.94
CNN Bi-LSTM	52.34	55.37	50.34	56.53
CNN Bi-LSTM + Sub-word	52.34	**56.82**	52.56	50.25
CNN Bi-LSTM All	**53.47**	56.75	**53.90**	**57.37**

Table 7. Comparison of various multilingual learning strategies and monolingual deep learning system on Tamil, Malayalam, Bengali and Marathi NER using Hindi as assisting language

	Tamil						**Marathi**					
	Mono			**Multi**			**Mono**			**Multi**		
	P	R	F	P	R	F	P	R	F	P	R	F
Person	62.38	52.65	57.10	**72.88**	47.91	**57.82**	53.68	18.75	27.79	**61.90**	**23.90**	**34.48**
Location	65.91	41.43	50.88	62.38	**46.19**	**53.08**	57.53	61.51	59.45	**64.84**	56.09	**60.15**
Organization	-	-	-	-	-	-	-	-	-	**20.00**	**2.86**	**5.00**
Overall	64.02	44.27	52.34	**66.67**	**44.63**	**53.47**	57.37	55.71	56.53	**64.56**	51.62	**57.37**

Table 8. Precision (P), Recall (R) and F-Score (F) (in percentage) for multilingual learning (All configuration) and monolingual deep learning system on Tamil and Marathi NER

	Malayalam						**Bengali**					
	Mono			**Multi**			**Mono**			**Multi**		
	P	R	F	P	R	F	P	R	F	P	R	F
Person	56.56	51.86	54.11	**67.53**	51.42	**58.39**	48.61	27.34	35.00	40.82	**31.25**	**35.40**
Location	60.10	60.00	60.05	**66.67**	53.11	59.12	64.08	46.41	53.83	**71.34**	**49.22**	**58.25**
Organization	-	-	-	**38.46**	**18.07**	**24.59**	33.33	1.82	3.45	33.33	**3.64**	**6.56**
Overall	58.66	52.43	55.37	**65.75**	49.91	**56.75**	62.52	42.13	50.34	**67.14**	**45.02**	**53.90**

Table 9. Precision (P), Recall (R) and F-Score (F) (in percentage) for multilingual learning (All configuration) and monolingual deep learning system on Malayalam and Bengali NER

Additionally, we also report results for Spanish and Dutch as primary languages with English as assisting language. The results are presented in Table 6. We observe sharing all layers to be beneficial for both Spanish and Dutch compared to sharing only sub-word features.

5.2 Low-Resource Language Results

Table 7 presents the results from our multilingual learning system (*CNN Bi-LSTM All*) and the comparison with the baseline systems for Indian languages *viz* Tamil, Malayalam, Bengali and Marathi. In the monolingual learning setup, the *CNN Bi-LSTM* model [28] outperforms CRF system on 3 out of 4 languages. However, the *CNN Bi-LSTM All* approach proposed here outperforms both these monolingual systems. Also, it is observed that our *CNN Bi-LSTM MLL* approach outperforms *CNN Bi-LSTM Sub-word* method on 3 out of 4 languages, while the performance gain obtained from *CNN Bi-LSTM Sub-word* model on the 4th language (Malayalam) is minimal. Further, it is observed that adding Hindi as the assisting language, improves NER performance for all the four languages. We observe an increase in F-Score of at least 0.8 absolute points from *CNN Bi-LSTM All* approach over the baseline monolingual systems.

The Tables 8 and 9 presents tag-wise results from both *CNN Bi-LSTM model* (mono) and *CNN Bi-LSTM All* model (multi) for all the languages. Here *P* stands for *Precision*, *R* for *Recall* and *F*

Approach	**Tamil**	**Malayalam**	**Bengali**	**Marathi**
CRF + POS	44.60	48.70	52.44	44.94
CNN Bi-LSTM	52.34	55.37	50.34	56.53
CNN Bi-LSTM Sub-word	52.34	56.82	52.56	50.25
CNN Bi-LSTM All	53.47	56.75	53.90	57.37
CNN Bi-LSTM + POS	58.65	61.25	52.46	**71.50**
CNN Bi-LSTM All + POS	**60.52**	**64.27**	**55.57**	70.90

Table 10. Comparison of various multilingual learning strategies and monolingual deep learning system on Tamil, Malayalam, Bengali and Marathi NER using Hindi as assisting language

for *F-Score*. In general, we observe improvements in both precision and recall. Specifically, the multilingual model was able to identify organization entities better compared to the monolingual model. *Organization* entities tends to be a challenge for both the monolingual and multilingual models.

Augmenting Features to Multilingual Learning

It is observed that by multilingual learning we are able to address data sparsity and improve the performance of the NER systems. Now, we would like to see if language independent features can be added to the multilingual learning system. Subsequently, we add POS features as additional feature to the deep learning system. The POS tag of a word is sent through a lookup table to obtain POS embeddings of dimension 10 in all our experiments. The obtained POS embedding is concatenated with the word embeddings and the sub-word features, before it is sent to the Bi-LSTM layer. Like our earlier experiment, we share POS lookup table and all the remaining layers between languages. We consider the *CNN Bi-LSTM* model with POS features as baseline system for comparison.

The Table 10 compares the performance of *CNN Bi-LSTM* model and *CNN Bi-LSTM All* model augmented with POS features. It is observed that by sharing POS features across languages there are improvements in 3 out of 4 languages. However, there is a slight drop in the performance for Marathi when POS features are added.

Hence, we can say that, multilingual Learning complements language independent features leading to further improvements in NER performance.

6 ANALYSIS

In this section we present a detailed analysis of the results from multilingual learning models.

6.1 Qualitative Analysis

We observe several instances where assisting language helps improve the named entity tagging performance of the primary language. We now present some instances from the test data where multilingual learning (*CNN Bi-LSTM All*) was found to be beneficial.

Transfer of Named Entity Vocabulary

By adding training data from an assisting language, we are increasing the named entity vocabulary seen by the model. This should benefit the primary language in the multilingual learning setting especially when the primary language has very less training data.

The Marathi training data does not contain any instance of the word न्यूयॉर्क (*newyork*). As a result, the monolingual model tags only 1 out of 5 instances in the test data correctly as *Location* entity. However, the word न्यूयॉर्क (*newyork*) is observed in Hindi training data. In this case, the *CNN Bi-LSTM All* model correctly tags all the 5 instances in the test data because of the knowledge transfer from the Hindi training data.

Transfer of Sub-word Features

The Marathi training data does not contain any instance of the word रॉबर्टो (*roberto*) or any of it's inflected forms. But, the Hindi training data contains the un-inflected form as रॉबर्ट (*robert*). As a result, the *CNN Bi-LSTM All* was able to correctly tag the word रॉबर्टो (*roberto*) as *Person* entity.

The Bengali training data contains the suffix चन्द्र (*chandra*) appearing 7 times. Two out of seven such words were tagged as *Location* entity, one instance was tagged as *Person* entity and the remaining as non-named entity. The same suffix चन्द्र (*chandra*) appears in the Hindi training data 5 times, twice as *Location* entity, twice as *Person* entity and once as non-named entity. Due to our multilingual learning approach, the confidence in this sub-word feature increased, and the model was able to tag words containing this suffix correctly. Similar patterns are observed for other language pairs where the assisting language training data helps in improving NER performance of the primary language.

Multilingual Learning as Regularizer

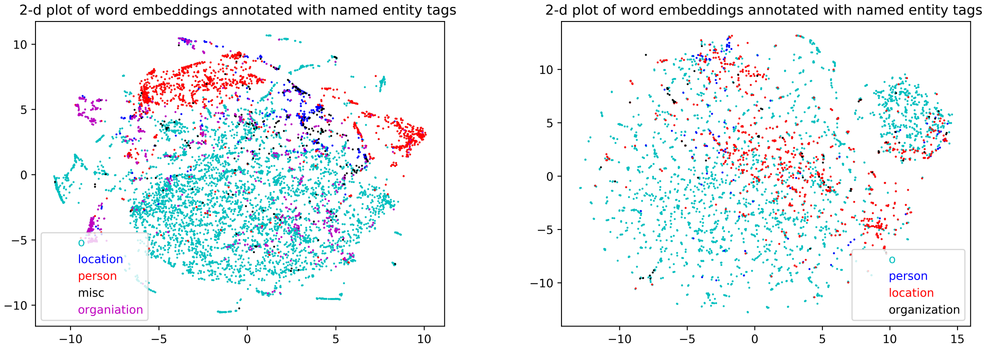
We observe instances where our Multilingual Learning strategy acts as regularizer. For example, the word हिन्दुस्थान् (*hindustan*)[3] appearing in the test data of Malayalam, gets tagged once as *Location* entity and another time as *Person* entity. The correct tag in both the cases is *Organization* entity. This word has the following left contextual pattern '*[number]* 'in both the sentences. On manual inspection, neither the word embedding features nor the sub-word features were found to be informative. We now look at the influence of contextual patterns in entity tagging. We obtained the Bi-LSTM layer output (instance-level representation) for this word in both the test sentences using both CNN Bi-LSTM and *CNN Bi-LSTM All* models. Similarly, we obtained instance-level representation for all words in the training data using *CNN Bi-LSTM* model as well as *CNN Bi-LSTM All* model. When we observed the nearest neighbors for this word in the instance-level representation space obtained from *CNN Bi-LSTM* model, we found all the nearest neighbors had the same left contextual pattern '*[number]* 'in the Malayalam training data. The Malayalam training data contains the pattern '*[number]*'followed by a *Person* or *Location* entity. The monolingual model memorizes this pattern and because of the tag ambiguity, tagged one occurrence as *Person* entity and the other occurrence as *Location* entity.

The *CNN Bi-LSTM All* model however, was able to correctly tag both the occurrence as *Organization* entity. Here, we hypothesize that our multilingual model was forced to look at better contextual clues and the reliance on the left contextual pattern '*[number]* 'was weighted down. Thus, multilingual learning played the role of regularization and helped our *CNN Bi-LSTM All* model to obtain better tagging performance.

Negative Influence of Assisting Language Data

There are also cases where multilingual learning negatively impacts the performance of the NER system. Marathi training data is mostly from the Tourism genre. We observe several instances of the

[3] The Malayalam data is converted to Devanagari Script using Indic NLP Library



(a) t-SNE map of English word embeddings

(b) t-SNE map of Bengali word embeddings

Fig. 5. 2d Visualization of word embeddings

location named entity को लांटा (*ko lanta*) in the Marathi test data. The *CNN Bi-LSTM* model is able to correctly tag 51 out of 56 instances for the word को (*ko*) as part of Location entity in the Marathi test data. In Hindi data, the word को (*ko*) appears more frequently in the role of post-position and not being a part of any named entity. However, because of the influence from the Hindi training data, the *CNN Bi-LSTM All* model correctly tags only 10 instances out of 56 as *Location* entity.

Quality of Word Embeddings

One of the primary reasons for the poor results from all the deep learning models is because of the quality of word embeddings. To validate the claim that word embeddings are poor for Indian languages, we plot a 2d-map of the word embeddings using t-SNE [40]. We obtained the monolingual Spectral word embeddings [6] for English and the pre-trained monolingual Fasttext word embeddings [2] for Bengali. We plot the 2-d map for all the words appearing in the English CoNLL 2003 Shared task test data and Bengali test data. We take the most frequent tag as the named entity tag for the word.

The Figure 5a and 5b shows the 2-d plot for English and Bengali respectively. We observe that in case of both language word embeddings, the named entities tend to appear together. English word embeddings form more coherent clusters where only named entities of the same type occur. This is not the case for Bengali word embeddings. The named entities in Bengali appear together in the embedding space, however they are accompanied by lot of non-named entity words too. Similar pattern is observed for the other Indian languages considered in our experiments. As a consequence, the results for Indian language NER are generally poor.

Influence of Named Entities Frequency

We observe that a large number of misclassified named entities either do not appear during training or appear infrequently in the training data. We obtain a list of misclassified named entities in the Tamil test set. We consider only those named entities which appear in the Tamil training data. As the model has encountered these named entities during training, the error on such named entities should be minimum. We order the misclassified named entities by their frequencies in the training data. The Figure 6 shows the bar plot of misclassified named entities ordered by frequency obtained from the Tamil test data. We restrict the frequency of words to 100, as misclassified words appearing more than 100 times in the training data are '*comma* (,), *parenthesis* (( )), *hyphen* (-) '. We observe

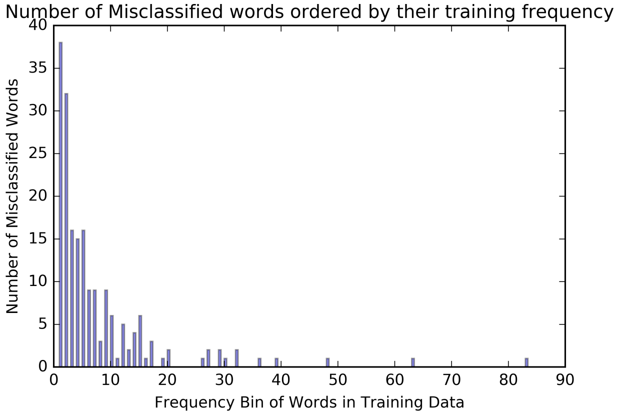


Fig. 6. Number of Misclassified words ordered by their training frequency in Tamil Test Data

a large number of misclassified named entities appear very rarely in the training data leading to lower NER performance.

6.2 Empirical Analysis

Influence of Assisting Language Corpus Size on Multilingual Learning

Here, we vary the percentage of assisting language corpus added to the primary language data and study it's influence on primary language tagging performance. We consider German as the primary language with English or Spanish or Dutch as the assisting languages. We vary the percentage of assisting language sentences added from 10% to 100% in increments of 10% with, the entire primary language data being considered. The results are as shown in the Table 11. We compare the performance of the multilingual learning model (sub-word and all configuration) with monolingual model. The last row indicates the performance of the monolingual baseline system trained using only primary language data.

We observe consistent improvements from both *Sub-word* and *All* configurations of multilingual learning over monolingual performance. For Spanish, the improvements from sharing only sub-word features are relatively lower compared to sharing all layers of the network. In general, multilingual learning by sharing all layers of the network gives better tagging performance compared to sharing only sub-word features across languages.

Can Assisting Language be Arbitrarily Chosen?

In all our multilingual experiments, we had chosen Hindi as the assisting language and tried to improve the performance of the other primary languages. As Hindi had comparatively larger training data, we chose Hindi as the assisting language. Now, the aim of this experiment is to find out if assisting language can be arbitrarily chosen or not.

The experimental setup is similar to our earlier configuration. The main difference being Hindi is the primary language and other languages are chosen to be the assisting languages. We choose the same hyper-parameter values for this experiment. We do not employ oversampling methods as Hindi has larger training data. However, we keep the value of β to 0.1 indicating the importance given to the objective of assisting language.

The Table 12 shows the effect of multilingual learning for Hindi. As we obtain bilingual embeddings for each pair of languages involved, we observe slightly different performances in the

Assisting Language	English		Spanish		Dutch	
Sentences (in %)	Sub-word	All	Sub-word	All	Sub-word	All
10	86.78	**87.20**	77.61	**83.81**	84.41	**84.78**
20	86.90	**87.20**	81.56	**84.98**	84.51	**85.71**
30	**86.37**	86.28	79.31	**84.52**	84.47	**85.80**
40	86.20	**86.96**	82.21	**85.14**	83.48	**87.27**
50	85.31	**87.33**	81.11	**85.31**	82.00	**87.89**
60	85.28	**87.40**	82.96	**85.42**	**86.40**	86.31
70	84.31	**86.62**	80.80	**84.45**	82.41	**87.32**
80	86.69	**88.16**	79.11	**85.67**	83.55	**85.44**
90	**88.08**	87.51	82.43	**85.46**	84.31	**85.69**
100	86.07	**88.18**	84.06	**85.74**	83.97	**87.78**
Monolingual Result	84.86		77.12		82.57	

Table 11. Influence of Assisting Language Corpus size on Multilingual learning (German Test F-Score)

Assisting Language	Monolingual	Multilingual
Malayalam	**61.59**	59.03
Marathi	**62.57**	57.35
Bengali	**60.56**	55.09
Tamil	**61.92**	53.74

Table 12. Hindi Test F-Score(%): Comparison of Monolingual vs Multilingual Learning on Hindi NER using Malayalam, Marathi, Bengali and Tamil as assisting languages

monolingual setting for Hindi. We observe a negative impact on the Hindi results because of multilingual learning.

In the multilingual learning objective defined in the Section 3.1 *i.e.,* equation 4, $P(D_P)$ is the primary language objective (Hindi) and $P(D_A)$ is the assisting language objective (Malayalam, Marathi, Bengali, Tamil). Ideally, we will have to choose examples from the assisting language such that we maximize the likelihood (or minimize the error) of the primary language data. However, in all our experiments, we have considered entire assisting data during training. As a consequence of this, some of the assisting data samples might add more confusion, thereby increasing the error on the primary language data. One way to tackle this issue is by setting the value of β to a lower value, however, setting the value too low implies that the assisting language is essentially not sharing any features.

Because of the noise introduced by the assisting language, we do not observe any improvements for Hindi, when Hindi is jointly trained with any of the remaining languages. While, this was not the case when we chose Hindi as the assisting language. Hence, we can say that adding Hindi as an assisting language brought in more informative cross-lingual features to the remaining languages, leading to overall improvement in NER performance.

Approach	Tamil		Malayalam		Bengali	
	Devanagari	Original	Devanagari	Original	Devanagari	Original
CNN Bi-LSTM	52.34		55.37		50.34	
CNN Bi-LSTM Sub-word	**52.34**	52.10	**56.82**	56.30	**52.56**	51.97
CNN Bi-LSTM All	**53.47**	48.35	**56.75**	54.02	**53.90**	52.67

Table 13. Effect of Character Script on Multilingual Learning

6.3 Effect of Script-Conversion on Multilingual Learning

In our experiments, we converted the scripts of Bengali, Tamil and Malayalam to Devanagari script. The intuition is having a common script across languages allows the shared sub-word layer to learn cross-lingual sub-word features. This is not the case when the scripts is different across languages.

The Table 13 reports the results without and with script conversion for multilingual learning. We observe that having a common script across languages outperforms multilingual learning without script conversion. In some cases, we observe improvements from multilingual learning without script conversion over monolingual model. This could be attributed to the regularization effect played by multilingual learning and use of bilingual embeddings.

7 CONCLUSION AND FUTURE WORK

In this paper, we show that multilingual learning improves NER performance for a low-resource language when jointly trained with a closely-related language. The key idea is maximal sharing of cross-lingual features by sharing all of the network layers across languages. We observe improvements in NER performance for Bengali, Tamil, Malayalam and Marathi when jointly trained with Hindi. For closely-related languages, we observe it is better to share all layers of the network than to share a subset of the layers of the network for NER task.

Multilingual learning strategy also complements other language independent features, where, we had shared POS features across languages and observed improvements in NER tagging. Multilingual learning can be applied to any low-resource language with the only constraint being availability of training data in a closely-related resource-rich language and access to bilingual word embeddings. As a part of future work, we would like to experiment our Multilingual learning approach with non-Indian languages.

Our experiments were mainly on language pairs or specifically bilingual learning. This approach could be extended to multiple languages, where we hope to improve NER tagging performance of the primary language by bringing in training data from multiple assisting languages. A major bottleneck is the availability of multilingual embeddings *i.e.,* similar words across multiple languages have similar embeddings in the common space.

The choice of assisting language is crucial as observed from our experiments involving Hindi as the primary language. A better strategy is to selectively choose only those sentences from the assisting language for training which helps reduce the error on the primary language data. As a future work, we would like to look at strategies for selecting only informative sentences from the assisting language, thereby decoupling the strong dependence on the choice of the assisting language in our model.

REFERENCES

[1] Pushpak Bhattacharyya. 2010. IndoWordNet. In *LREC.*

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *TACL* (2017).

[3] Yufeng Chen, Chengqing Zong, and Keh-Yih Su. 2013. A Joint Model to Identify and Align Bilingual Named Entities. *Computational Linguistics* 39, 2 (June 2013), 229–266. https://doi.org/10.1162/COLI_a_00122

[4] Jason Chiu and Eric Nichols. 2016. Named Entity Recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics* 4 (2016).

[5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *The Journal of Machine Learning Research* 12 (Nov. 2011), 2493–2537. http://dl.acm.org/citation.cfm?id=1953048.2078186

[6] Paramveer S. Dhillon, Dean P. Foster, and Lyle H. Ungar. 2015. Eigenwords: Spectral Word Embeddings. *Journal of Machine Learning Research* 16 (2015), 3035–3078. http://jmlr.org/papers/v16/dhillon15a.html

[7] Cicero dos Santos, Victor Guimaraes, RJ Niterói, and Rio de Janeiro. 2015. Boosting Named Entity Recognition with Neural Character Embeddings. *Proceedings of NEWS 2015 The Fifth Named Entities Workshop* (2015).

[8] Cícero Nogueira Dos Santos and Bianca Zadrozny. 2014. Learning Character-level Representations for Part-of-speech Tagging. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, II–1818–II–1826. http://dl.acm.org/citation.cfm?id=3044805.3045095

[9] Manaal Faruqui. 2014. "Translation can't change a name": Using Multilingual Data for Named Entity Recognition. abs/1405.0701 (2014). arXiv:1405.0701 http://arxiv.org/abs/1405.0701

[10] Manaal Faruqui and Sebastian Padó. 2010. Training and Evaluating a German Named Entity Recognizer with Semantic Generalization. In *Proceedings of KONVENS 2010*. Saarbrücken, Germany.

[11] Orhan Firat, Kyunghyun Cho, Baskaran Sankaran, Fatos T. Yarman Vural, and Yoshua Bengio. 2017. Multi-way, Multilingual Neural Machine Translation. *Computer Speech and Language* 45, C (Sept. 2017), 236–252. https://doi.org/10.1016/j.csl.2016.10.006

[12] Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. 2016. Multilingual Language Processing From Bytes. *In proceedings of NAACL-HLT (NAACL 2016)*. (2016).

[13] Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. BilBOWA: Fast Bilingual Distributed Representations without Word Alignments. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, David Blei and Francis Bach (Eds.). JMLR Workshop and Conference Proceedings, 748–756.

[14] James Hammerton. 2003. Named Entity Recognition with Long Short-term Memory. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*. Association for Computational Linguistics.

[15] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. abs/1508.01991 (2015). arXiv:1508.01991 http://arxiv.org/abs/1508.01991

[16] Girish Nath Jha. 2012. The TDIL program and the Indian Language Corpora Initiative. In Language Resources and Evaluation Conference.

[17] Melvin Johnson, Mike Schuster, Quoc Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda ViÃľgas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics* (2017).

[18] Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*. AAMT, AAMT, Phuket, Thailand, 79–86.

[19] Michal Konkol and Miloslav Konopík. 2013. CRF-Based Czech Named Entity Recognizer and Consolidation of Czech NER Research. In *Text, Speech, and Dialogue*, Ivan Habernal and Václav Matoušek (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 153–160.

[20] Anoop Kunchukuttan and Pushpak Bhattacharyya. 2016. Orthographic Syllable as basic unit for SMT between Related Languages. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*.

[21] Anoop Kunchukuttan, Ratish Puduppully, and Pushpak Bhattacharyya. 2015. Brahmi-Net: A transliteration and script conversion system for languages f the Indian subcontinent. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. Association for Computational Linguistics.

[22] Shobha Lalitha Devi, Pattabhi RK Rao, Malarkodi C.S, and R Vijay Sundar Ram. 2014. Indian Language NER Annotated FIRE 2014 Corpus (FIRE 2014 NER Corpus). In Named-Entity Recognition Indian Languages FIRE 2014 Evaluation Track.

[23] Guillaume Lample, Miguel Ballesteros, Kazuya Kawakami, Sandeep Subramanian, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *In proceedings of NAACL-HLT (NAACL 2016)*. San Diego, US.

[24] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Chris Bizer. 2014. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal* (2014).

[25] Qi Li, Haibo Li, Heng Ji, Wen Wang, Jing Zheng, and Fei Huang. 2012. Joint Bilingual Name Tagging for Parallel Corpora. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM*

'12). ACM, New York, NY, USA, 1727–1731. https://doi.org/10.1145/2396761.2398506

[26] L. Liu, J. Shang, F. Xu, X. Ren, H. Gui, J. Peng, and J. Han. 2018. Empower Sequence Labeling with Task-Aware Neural Language Model. In *AAAI*.

[27] Xuezhe Ma and Eduard Hovy. 2016. 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)*. Association for Computational Linguistics, Berlin, Germany.

[28] Rudra V. Murthy and Pushpak Bhattacharyya. 2016. A Complete Deep Learning Solution to Named Entity Recognition. In *CICLing 2016*. Konya, Turkey.

[29] Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

[30] Lev Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL '09)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 147–155. http://dl.acm.org/citation.cfm?id=1596374.1596399

[31] Marek Rei. 2017. Semi-supervised Multitask Learning for Sequence Labeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[32] M. Schuster and K.K. Paliwal. 1997. Bidirectional Recurrent Neural Networks. *Trans. Sig. Proc.* 45, 11 (Nov. 1997), 2673–2681. https://doi.org/10.1109/78.650093

[33] Anil Kumar Singh. 2008. Named Entity Recognition for South and South East Asian Languages: Taking Stock.

[34] Raivis Skadiņš, Jörg Tiedemann, Roberts Rozis, and Daiga Deksne. 2014. Billions of Parallel Words for Free: Building and Using the EU Bookshop Corpus. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC-2014)*. European Language Resources Association (ELRA), Reykjavik, Iceland.

[35] Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *ICLR* (2017).

[36] Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 Shared Task: Language-independent Named Entity Recognition. In *Proceedings of the 6th Conference on Natural Language Learning - Volume 20 (COLING-02)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 1–4. https://doi.org/10.3115/1118853.1118877

[37] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4 (CONLL '03)*. Association for Computational Linguistics.

[38] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word Representations: A Simple and General Method for Semi-supervised Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 384–394. http://dl.acm.org/citation.cfm?id=1858681.1858721

[39] Kãrumũri V Subbãrão. 2012. South Asian Languages: A Syntactic Typology. (2012).

[40] L.J.P. van der Maaten and G.E. Hinton. 2008. Visualizing High-Dimensional Data Using t-SNE. (2008).

[41] Mengqiu Wang, Wanxiang Che, and Christopher D. Manning. 2013. Effective Bilingual Constraints for Semi-supervised Learning of Named Entity Recognizers. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI'13)*. AAAI Press, 919–925. http://dl.acm.org/citation.cfm?id=2891460.2891588

[42] Mengqiu Wang, Wanxiang Che, and Christopher D. Manning. 2013. Joint Word Alignment and Bilingual Named Entity Recognition Using Dual Decomposition. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*.

[43] Mengqiu Wang and Christopher D. Manning. 2014. Cross-lingual Projected Expectation Regularization for Weakly Supervised Learning. *TACL* (2014).

[44] Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2017. Multi-Task Cross-Lingual Sequence Tagging from Scratch. *ICLR* (2017).