Multilingual Code-switching Identification via LSTM Recurrent Neural Networks

Younes Samih[†] Suraj Mahrjan[‡]
Mohammed Attia[◇] Laura Kallmeyer[†] Thamar Solorio[‡]

University of Düsseldorf[†] *Houston University*[‡] *Google Inc.*[◇]

EMNLP 2016 Second Workshop on Computational Approaches to Code Switching
Austin, Texas USA November, 1, 2016

Content

- Linguistic Background

Content

- Linguistic Background
- Dataset

Content

- Linguistic Background
- Dataset
- Neural Network

Content

- Linguistic Background
- Dataset
- Neural Network
- Approach

Content

- Linguistic Background
- Dataset
- Neural Network
- Approach
- Summary

Code-switching

Linguistic Background

- speakers switch from one language or dialect to another within the same context [Bullock and Toribio, 2009]
- Three types of codes-switching: inter-sentential, Intra-sentential, intra-word

Code-switching

Linguistic Background

- speakers switch from one language or dialect to another within the same context [Bullock and Toribio, 2009]
- Three types of codes-switching: inter-sentential, Intra-sentential, intra-word

Constraints on Code-switching

- equivalence constraint [Poplack 1980]
- The Matrix Language-Frame (MLF)[Myers-Scotton 1993]
  - Matrix language (ML)
  - The embedded language (EL)

Shared Task Dataset

MSA-Egyptian Data

	all	training	dev	test
tweets	11,241	8,862	1,117	1,262
tokens	227,329	185,928	20,688	20,713

Table: MSA-Egyptian Data statistics

Spanish-English Data

	all	training	dev	test
tweets	21,036	8,733	1,587	10,716
tokens	294,261	139,539	33,276	121,446

Table: Spanish-English Data statistics

Corpora

Arabic Corpus

genre	tokens
Facebook posts	8,241,244
Tweets	2,813,016
News comments	95,241,480
MSA news texts	276,965,735
total	**383,261,475**

Table: Arabic corpus statistics

Spanish-English Corpus

- English gigaword corpus(Graff et al.,2003)
- Spanish gigaword corpus (Graff ,2006)

Data preprocessing

Data preprocessing

- mapping Arabic scripts to SafeBuckwalter
- conversion of all Persian numbers to Arabic numbers

Data preprocessing

Data preprocessing

- mapping Arabic scripts to SafeBuckwalter
- conversion of all Persian numbers to Arabic numbers
- conversion of Arabic punctuation to Latin punctuation
- remove kashida (elongation character) and vowel marks

Data preprocessing

Data preprocessing

- mapping Arabic scripts to SafeBuckwalter
- conversion of all Persian numbers to Arabic numbers
- conversion of Arabic punctuation to Latin punctuation
- remove kashida (elongation character) and vowel marks
- separate punctuation marks from words

Neural network

- Recurrent Neural Network
- Long short-term memory network
- Word Embeddings

Reccurent Neural Network

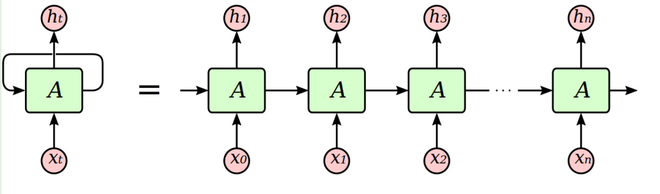


Figure by Christopher Olah

RNN

Given input sequence: $x_1, x_2, ..., x_n$

Reccurent Neural Network

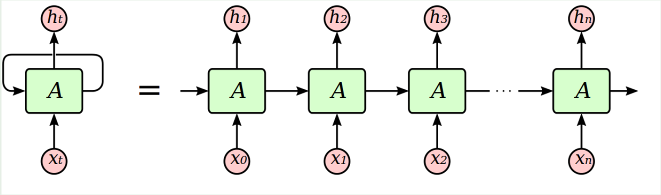


Figure by Christopher Olah

RNN

Given input sequence: $x_1, x_2, ..., x_n$

a standard RNN computes the output vector y_t of each word x_t

Reccurent Neural Network

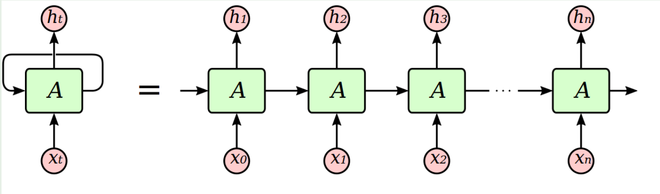


Figure by Christopher Olah

RNN

Given input sequence: $x_1, x_2, ..., x_n$

a standard RNN computes the output vector y_t of each word x_t

$$h_t = H(W_{x_h} x_t + W_{h_h} h_{-1} + b_h)$$

$$y_t = y_{h_y} + b_y$$

Long-term dependencies

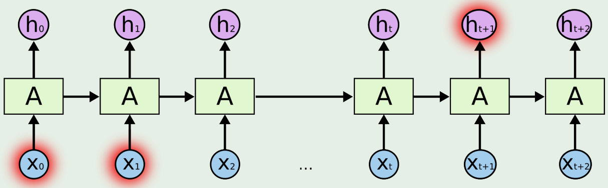


Figure by Christopher Olah

Long-term dependencies

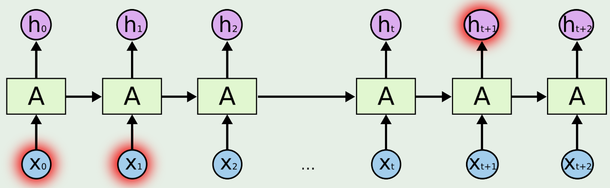


Figure by Christopher Olah

Basics

- Problem learning long-term dependencies in the data

Long-term dependencies

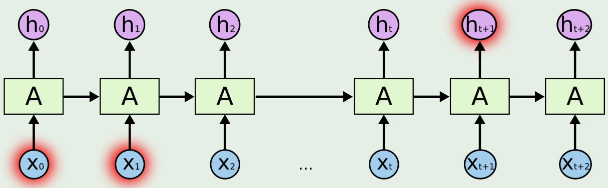


Figure by Christopher Olah

Basics

- Problem learning long-term dependencies in the data

Long-term dependencies

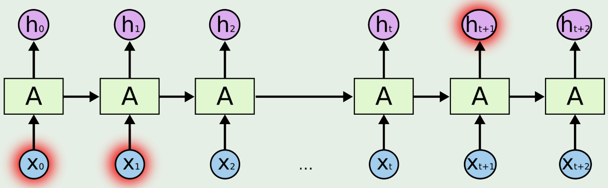


Figure by Christopher Olah

Basics

- Problem learning long-term dependencies in the data
- Vanishing gradients

Long-term dependencies

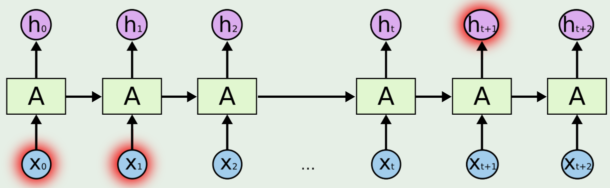


Figure by Christopher Olah

Basics

- Problem learning long-term dependencies in the data
- Vanishing gradients
- exploding gradients

Long short-term memory network

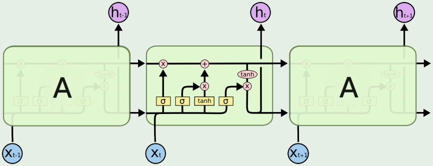


Figure by Christopher Olah

LSTM Basics

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C.[h_{t-1}, x_t] + b_C)$$
$$C_t = f_t.C_t - 1 + i_t.\tilde{C}_t$$
$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * \tanh(C_t)$$

Vector Space Models

- Vector space models
- Distributional hypothesis: Words in the same contexts share the same meaning
  - Count-based methods (Latent Semantic Analysis,...)
  - Neural probabilistic language models(Word embeddings)

Word2vec

- The main component of the neural-network approach

Word2vec

- The main component of the neural-network approach
- Representation of each feature as a vector in a low dimensional space

Word2vec

- The main component of the neural-network approach
- Representation of each feature as a vector in a low dimensional space
- Continuous Bag-of-Words model (CBOW) vs Skip-Gram model

Word Embeddings

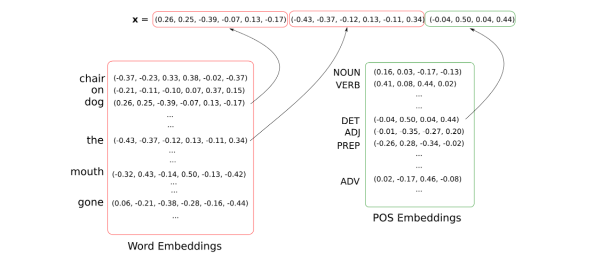


Figure by Yoav Goldberg

Code-switching detection

- System Architecture
- Implementation Details
- Results
- Summary

System Architecture

LSTM-CRF for Code-switching Detection

Our neural network architecture consists of the following three layers:

- Input layer: comprises both character and word embeddings

System Architecture

LSTM-CRF for Code-switching Detection

Our neural network architecture consists of the following three layers:

- Input layer: comprises both character and word embeddings
- Hidden layer: two LSTMs map both words and character representations to hidden sequences

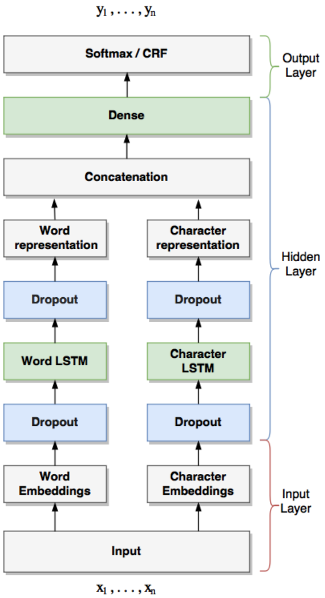
System Architecture

LSTM-CRF for Code-switching Detection

Our neural network architecture consists of the following three layers:

- Input layer: comprises both character and word embeddings
- Hidden layer: two LSTMs map both words and character representations to hidden sequences
- Output layer: a Softmax or a CRF computes the probability distribution over all labels

System Architecture



Implementation Details

- Pre-trained Word embeddings
- Character embeddings

Implementation Details

- Pre-trained Word embeddings
- Character embeddings
- Optimization: Dropout

Implementation Details

- Pre-trained Word embeddings
- Character embeddings
- Optimization: Dropout
- Output layer: Softmax or CRF

Implementation Details

- Pre-trained Word embeddings
- Character embeddings
- Optimization: Dropout
- Output layer: Softmax or CRF
- Training: Stochastic gradient descent
- optimizing Cross-entropy Objective function

Implementation Details

- Pre-trained Word embeddings
- Character embeddings
- Optimization: Dropout
- Output layer: Softmax or CRF
- Training: Stochastic gradient descent
- optimizing Cross-entropy Objective function
- Hyper-parameters tuning on Devset

Results on Spanish-English Dev set

Labels	CRF (feats)	CRF (emb)	CRF (feats+ emb)	word LSTM	char LSTM	char-word LSTM
ambiguous	0.00	0.02	0.00	0.00	0.00	0.00
fw	0.00	0.00	0.00	0.00	0.00	0.00
lang1	0.97	0.97	0.97	0.93	0.94	0.96
lang2	0.96	0.95	0.96	0.91	0.89	0.93
mixed	0.00	0.00	0.00	0.00	0.00	0.00
ne	0.52	0.51	0.57	0.34	0.13	0.32
other	1.00	1.00	1.00	0.85	1.00	1.00
unk	0.04	0.08	0.10	0.00	0.00	0.04
Accuracy	0.961	0.960	**0.963**	0.896	0.923	0.954

Table: F1 score results on Spanish-English development dataset

Results on MSA-Egyptian Dev set

Labels	CRF (feats)	CRF (emb)	CRF (feats+ emb)	word LSTM	char LSTM	char- word LSTM
ambiguous	0.00	0.00	0.00	0.00	0.00	0.00
lang1	0.80	0.88	0.88	0.86	0.57	0.88
lang2	0.83	0.91	0.91	0.92	0.23	0.92
mixed	0.00	0.00	0.00	0.00	0.00	0.00
ne	0.83	0.84	0.86	0.84	0.66	0.84
other	0.97	0.97	0.97	0.92	0.97	0.97
Accuracy	0.829	0.894	0.896	0.896	0.530	**0.900**

Table: F1 score results on MSA-Egyptian development dataset

Tweet level results

Scores	Es-En	MSA
Monolingual F1	0.92	0.890
Code-switched F1	0.88	0.500
Weighted F1	0.90	0.830

Table: Tweet level results on the test dataset.

Token level results

Label	Recall	Precision	F-score
ambiguous	0.000	0.000	0.000
fw	0.000	0.000	0.000
lang1	0.922	0.939	0.930
lang2	0.978	0.982	0.980
mixed	0.000	0.000	0.000
ne	0.639	0.484	0.551
other	0.992	0.998	0.995
unk	0.120	0.019	0.034
Accuracy			0.967

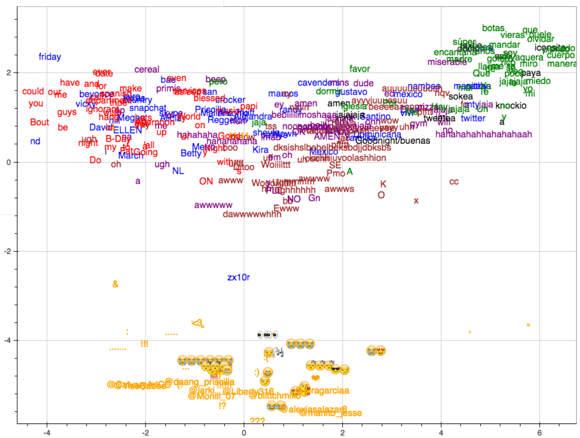
Table: Token level results on Spanish-English test dataset.

Token level results

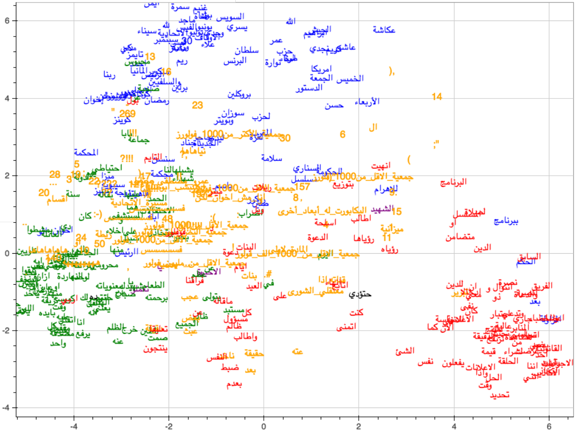
Label	Recall	Precision	F-score
ambiguous	0.000	0.000	0.000
fw	0.000	0.000	0.000
lang1	0.877	0.832	0.854
lang2	0.913	0.896	0.904
mixed	0.000	0.000	0.000
ne	0.729	0.829	0.777
other	0.938	0.975	0.957
unk	0.000	0.000	0.000
Accuracy			0.879

Table: Token level results on MSA-DA test dataset.

Spanish-English



MSA-Egyptian



CRF Model

Most likely	Score	Most unlikely	Score
unk ⇒ *unk*	1.789	*lang*1 ⇒ *mixed*	-0.172
ne ⇒ *ne*	1.224	*mixed* ⇒ *lang*1	-0.196
fw ⇒ *fw*	1.180	*amb* ⇒ *other*	-0.244
*lang*1 ⇒ *lang*1	1.153	*ne* ⇒ *mixed*	-0.246
*lang*2 ⇒ *lang*2	1.099	*mixed* ⇒ *other*	-0.254
other ⇒ *other*	0.827	*fw* ⇒ *lang*1	-0.282
*lang*1 ⇒ *ne*	0.316	*ne* ⇒ *lang*2	-0.334
other ⇒ *lang*1	0.222	*unk* ⇒ *ne*	-0.383
*lang*2 ⇒ *mixed*	0.216	*lang*2 ⇒ *lang*1	-0.980
*lang*1 ⇒ *other*	0.191	*lang*1 ⇒ *lang*2	-0.993

Table: Most likely and unlikely transitions learned by CRF model for the Spanish-English dataset.

Summary

- Automatic identification of code-switching in tweets
- A unified neural network for language identification
- rivals state-of-the-art methods that rely on language-specific tools

Summary

- Automatic identification of code-switching in tweets
- A unified neural network for language identification
- rivals state-of-the-art methods that rely on language-specific tools

What next?

- Implement character aware Bidirectional LSTM to capture word morphology
- Employ the More sophisticated CNN-Bidirectional LSTM

Thank you for your attention!

Questions?