Context-based Sentence Segmentation for Irregular Domains

Bachelor's Thesis

Krisztina Agoston

NLP pipeline

NLP = Natural Language Processing

Sentence segmentation:

- find the sentence boundaries in the input text
- divide the input text into individual sentences
- Formal definition:
  map the input text x to $y_1 \ldots y_n$ sentences

Sentence segmentation example

LORANSTA Marcus Island was billeted for 23 U.S. Coast Guard personnel. The commissioning commanding officer was U.S. Coast Guard Lieutenant Commander Louis. C. Snell.

Sentence Segmentizer

LORANSTA Marcus Island was billeted for 23 U.S. Coast Guard personnel.

The commissioning commanding officer was U.S. Coast Guard Lieutenant Commander Louis. C. Snell.

Complexity

Ambiguity examples:

- Abbreviation: *U.S. Special Operations Forces*
- Abbreviation at sentence end: *in Washington D.C.*
- Initials: *J. F. Kennedy*
- Ordinal numbers: *1. Section*
- Ellipses: *I have a dream....I have a dream today.*
- Quotes: *[...] 24 years later in his "Tear down this wall!" speech.*

How many punctuations do not denote a sentence end?

It depends...

- Brown: 12%
- Wall Street Journal: 42%

- Problem definition
- Approaches
- Machine Learning Solution
- Evaluation
- Conclusion

Approaches for sentence segmentation

Rule based

- Find sentence boundaries with predefined rules

Statistical

- Collect statistics from the text
- Calculate probabilities to disambiguate punctuation

Machine learning

- Use neural networks
- Train the automated system

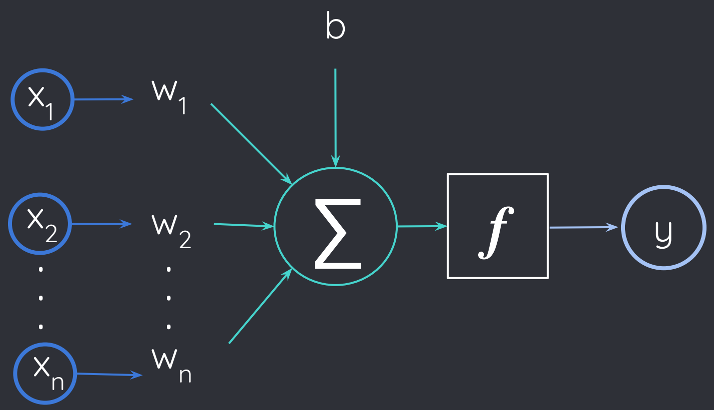
Machine learning in a Nutshell

Based on neural networks

Neurons (nodes) build up a network

Each neuron calculates $y = \sigma(\Sigma(w_i \cdot x_i) + b)$

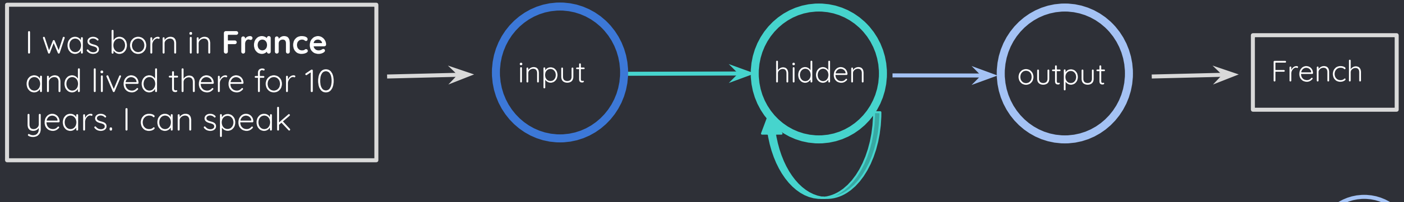
Goal: given the input $x_1 \dots x_n$ predict the output y



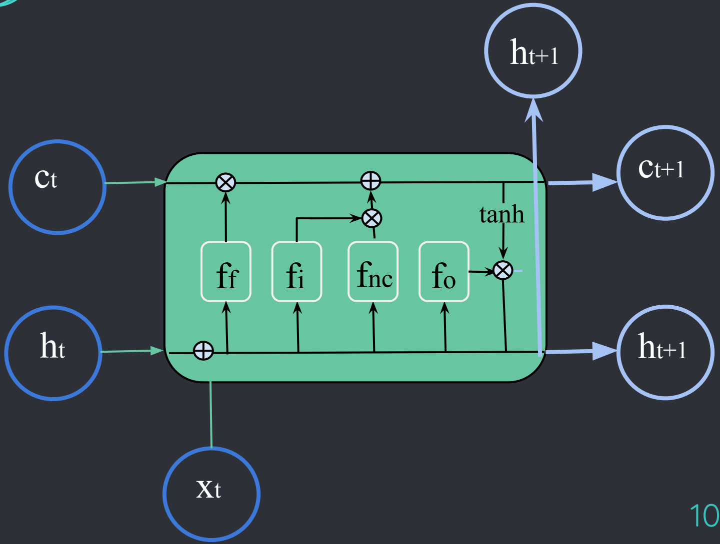
- i is the index of the sample
- $w_i \in \mathbb{R}$ is some weight to determine the input's importance
- bias $b \in \mathbb{R}$ is a constant value
- Nonlinear activation function f for creating probability $\sigma(z) = \frac{1}{1+e^{-z}}$ for the classes

Long Short-Term Memory Neural Network (LSTM)

- preferred to be used for sequential data
- capable of learning long-term dependencies



- distinguished by its gated structure:
  - Forget gate f_f
  - Input gate f_i
  - New candidate f_{nc}
  - Output gate f_o
- and its cell state c_t
  - = memory



Context windows

Character- or token-based

- For every unit in the text
- For every punctuation
- For every possible sentence end marks

The 35th U.S. President J. F. Kennedy died in 1963.

Window size = 6:

the		35th		u	.

died		in		1963	.

Labels: 0 1

Layers in the LSTM model / 1

Input layer provides data in the correct form

- context windows in batches

Embedding layer encodes the units

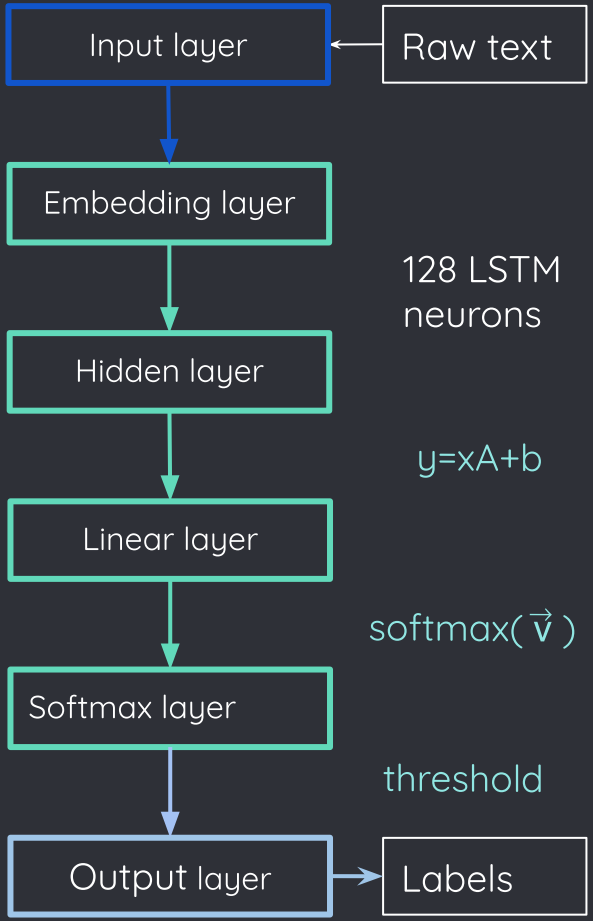
- Creating embedding vectors

LSTM layer

- perform a non-linear computation for each unit in a batch

Linear layer

- transforms the output of the hidden layer to the label space



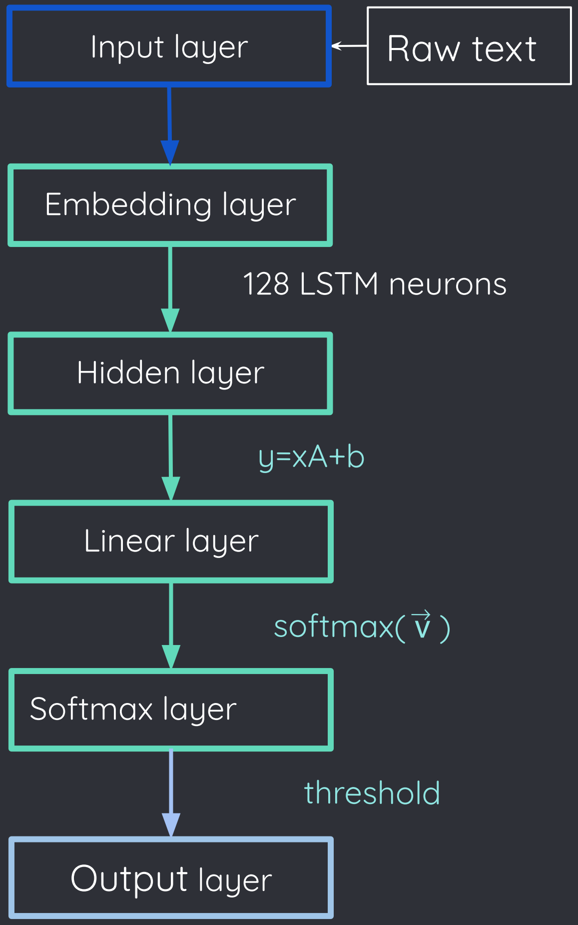
Layers in the LSTM model / 2

Softmax layer

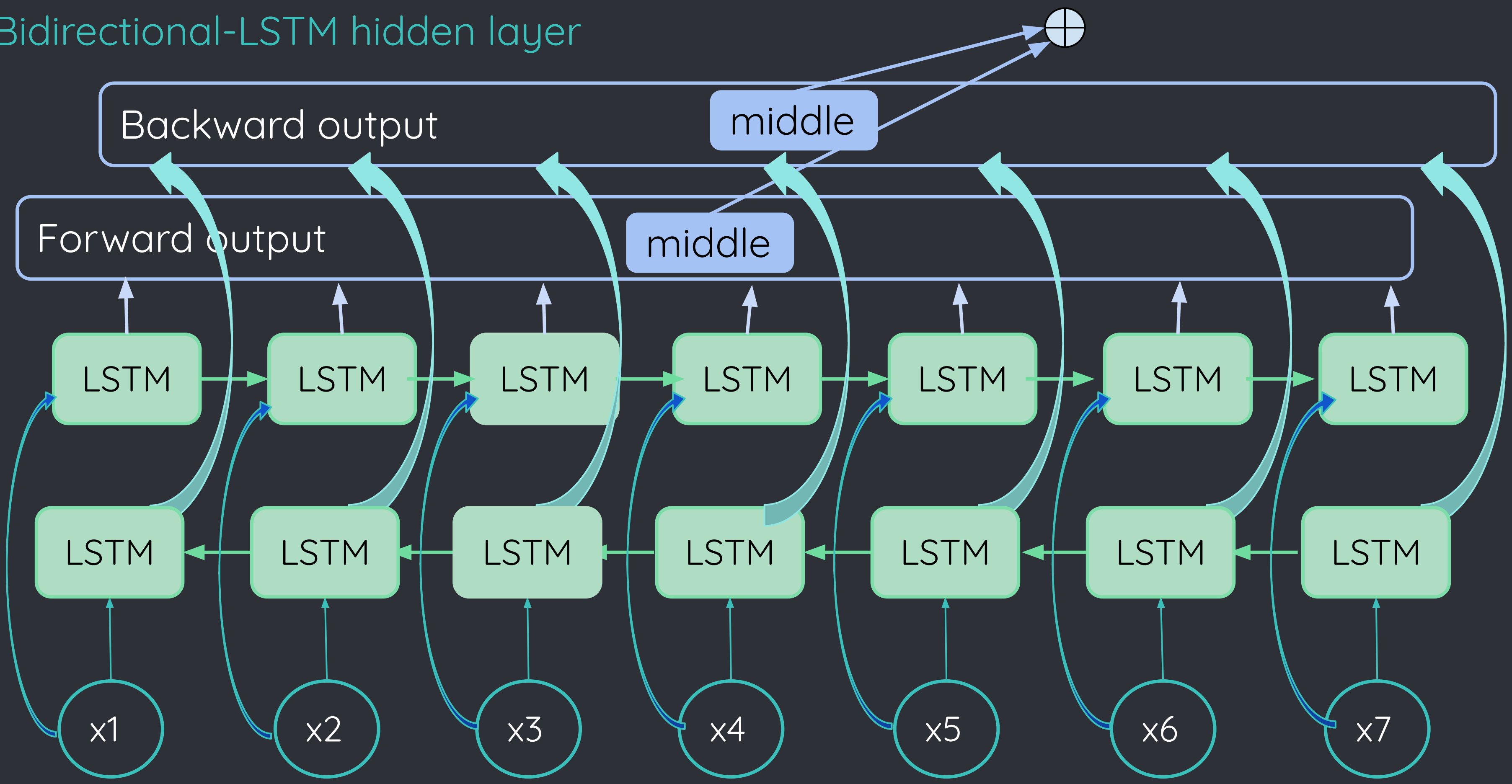
- transforms the real numbers within a vector into real numbers between 0 and 1 so that they sum up to 1
- $\text{softmax}(v) = \dfrac{e^{v_i}}{\sum_{i=1}^{n} (e^{v_i})}$
- We can use these values as probabilities

Output layer

- $\text{Label} \begin{cases} 1 & \text{if probability} \geq \text{threshold} \\ 0 & \text{otherwise} \end{cases}$



Bidirectional-LSTM hidden layer



Problem definition

Approaches

Machine Learning Solution

Evaluation

Conclusion

Domains

- Wall Street Journal (WSJ)
  - Sentences from the journal
- Brown
  - Works published in the USA in 1961
- Europarl
  - Extracted from the proceedings of the European Parliament
- Wikipedia
  - paragraphs from Wikipedia entries
- arXiv
  - paragraphs from scholarly articles from several research domain

Irregular example:
Z. Maki, M. Nakagawa, and S. Sakata, Prog. Theor. Phys. 28, 870 (1962).

V.N. 1992, Astroph. Space Phys., Sov. Sci. Rev. E, ed. R.A.Sunyaev 8,1 [...]

Datasets

Divided: 80% for training

- 10% for development
- 10% for final test

Training set from paragraphs:

- Take paragraphs shorter than 100 characters
- split paragraphs on *punctuation + whitespace + sentence starter (however, because, the, she)*
- Errors : sentence starts with a word not in our set punctuation followed by word from our set (not significant)

	sentences
WSJ	3.914
Brown	56.323
Europarl	1.906.966
	paragraphs
arXiv	1.006.228
Wikipedia	45.676.715

Test set for evaluation

Hard cases from the result of NLTK and spaCy segmentizers

Random cases

Size of the test subset with random sentences and hard cases

	hard cases		random	
corpus	paragraphs	sentences	paragraphs	sentences
Wall Street Journal			571	3.914
Europarl			1043	6.876
Brown			1.409	9995
arXiv	1.000	4.091		
Wikipedia	775	2.928	929	3.267

Metrics

$$\text{Error rate} = \frac{FP + FN}{FP + FN + TP}$$

$$\text{Precision} = \frac{TP}{FP + TP}$$

$$\text{Recall} = \frac{TP}{FN + TP}$$

$$\text{F1-score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

$$\text{Accuracy} = \frac{\text{Correct sentences}}{\text{All sentences}}$$

Predicted

Actual		Positive	Negative
	Positive	TP	FN
	Negative	FP	TN

Confusion matrix

True positive:
correctly predicted boundary

False positive:
no actual sentence boundary predicted as one

False negative:
missed sentence boundary

True negative: most of the cases
not a sentence boundary,
not predicted as a sentence boundary

Baseline systems

Baseline algorithm

- Define a lower bound
- Split on *punctuation + whitespace + capitalized letter*

```
r"([\.\?\!\"\)])\s+([A-Z])"
```

NLTK

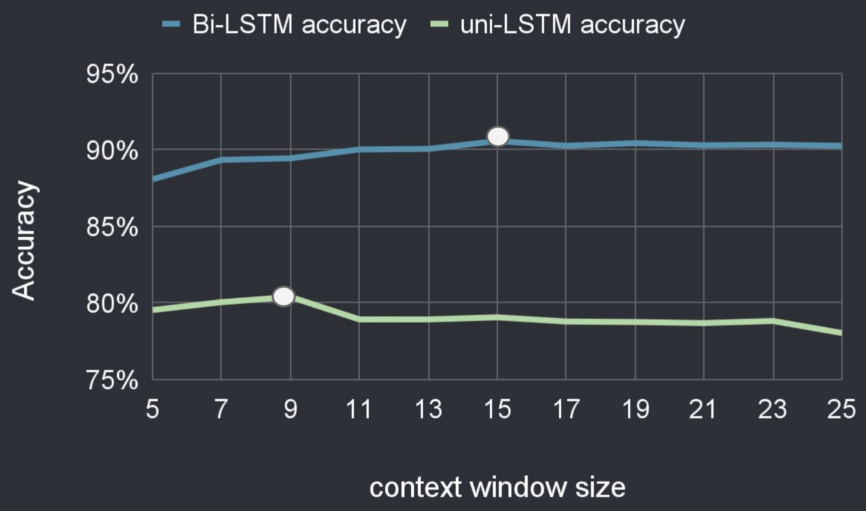
- Complete NLP toolkit
- Statistical segmentizer: Punkt sentence tokenizer
- Trainable with unsupervised training

spaCy

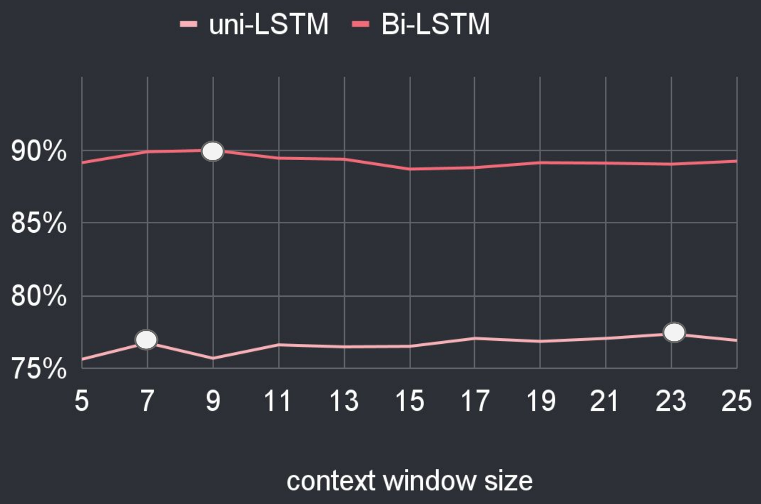
- State-of-the-art system with diverse components
- Senter trainable neural network-based segmentizer

Accuracy by growing context size

Character-based models



Token-based models

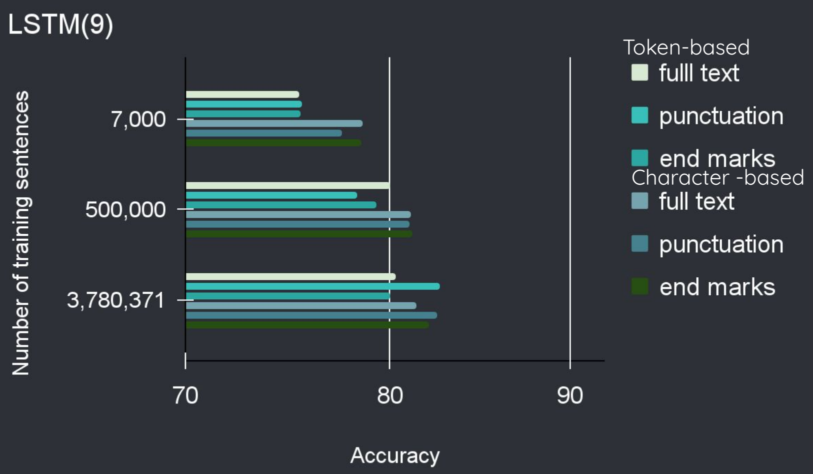


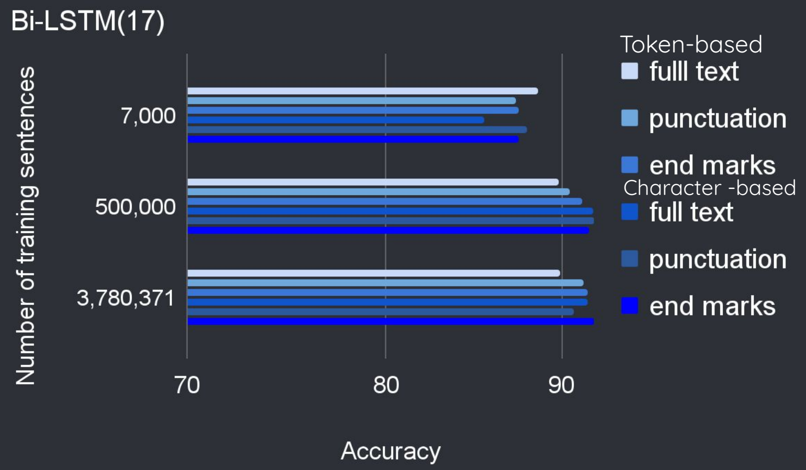
context window created only for punctuation, trained for 10 epochs on 500,000 Wikipedia sentences

- The larger context improves accuracy values by both LSTM models
- Over the optimum the uni-LSTM's accuracy drops

Training data size

Model trained on Wikipedia corpus,, evaluated on the development set (2753 sentences with hard cases)
Small dataset trained for 100 epochs, large for 10 epochs





Larger training data improves accuracy, especially by LSTM

Character-based models perform better

Comparison of accuracy of different systems on various corpora

Uni- and bi-STM models: character-based context window on punctuation, trained with early stopping

Corpus	Baseline	LSTM(9)	bi-LSTM(15)	default NLTK	trained NLTK	senter spaCy
Europarl	90.98	95.72	**96.80**	95.39	95.68	95.10
arXiv hard	92.30	90.37	**92.91**	74.31	87.29	91.93
Brown	67.43	92.17	**93.94**	85.01	83.91	93.84
Wikipedia hard	80.12	82.04	**90.64** *	84.19	89.07	79.44
Wikipedia rand.	91.86	93.44	96.94	**99.42**	97.86	95.56
WSJ	69.79	72.75	89.29	**92.79**	91.74	80.32

** bi-LSTM was trained on 500.000 sentences*

Bi-LSTM model outperforms other systems on most of the corpora

Cross-corpora evaluation: trained on Wikipedia, evaluated on WSJ

Review

Automated, trainable, easily adaptable solution for SBD	⟶ Neural Networks (NN)
Deal with sequential data	⟶ Special kind of NN: LSTM
Involve not only the previous inputs in the current calculation, but also the upcoming context	⟶ Bidirectional LSTM
Irregular domains	⟶ arXiv, Wikipedia

Conclusion

Approach:

- Character-based approach improves recall and decreases error rate

Context

- Using previous and next context improves precision by bi-LSTM, that outperforms other systems

Training:

- large training data to get an accurate model

Domains:

- The type of the domain determines the task-complexity

Goal of future work: general purpose model with improved runtime

Sources

- Speech and Language Processing.
  Daniel Jurafsky & James H. Martin. Copyright © 2021.
- Wong DF, Chao LS, Zeng X.
  iSentenizer-μ: multilingual sentence boundary detection model.
  ScientificWorldJournal. 2014

Extra slides

Rule-based approach
Statistical approach
Dependencies
Embeddings
Data preparation
Context window 1
Context window 2
F1 score
FB score

Baseline performance
Training time - context size
Systems on Wikipedia
Hard cases vs. random

Dimensionalities
Threshold
Training setup
Early stopping
Bi-LSTM

Common Errors - uni LSTM vs. Bi-LSTM

Unidirectional LSTM:

several punctuation marks next to each other

- It was as if he didn’t want the guests to be there. || ”
- Give the enemy no rest . ||.. Do all the

Period after a number

- or VBLANK interrupts on IRQ 5. This allows the use

Bidirectional LSTM

split after semicolon

- Sergio Ortega; || Leon Schidlowsky;

Common errors

False negative predictions are the missed sentence boundaries:

- .. translated as "Q īng Chéng" () The name...
- at all hazards. ... Give the enemy no rest ...
- "the sexiest man on TV." As Eddie films
- in Washington D.C. Before and

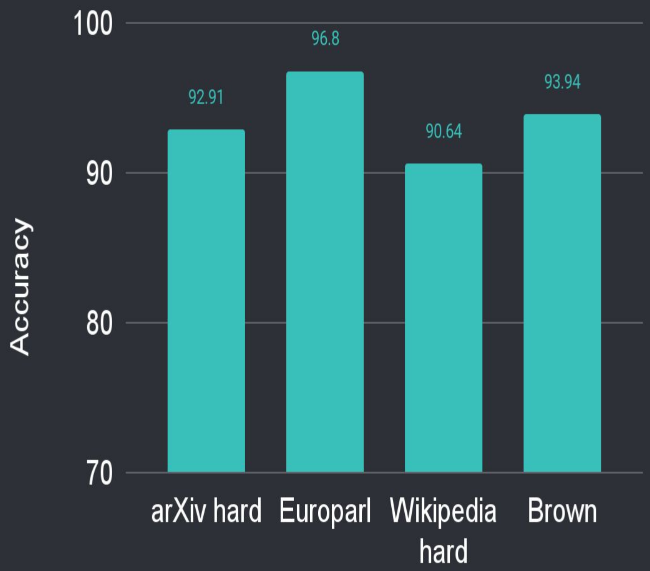
False positive ||

- "chenel" || "canal".
- "Do You Miss Me?" || became a Top 40 hit
- including Yahoo! Wireless in London, Splash! || in Germany,

Character based bi-LSTM(15) model evaluated on different corpora

Corpus	Error rate	Precision	Recall	F1-score	Accuracy
Context window on punctuations					
arXiv hard	0.05	0.98	0.97	0.97	92.91%
Europarl	0.02	0.99	0.98	0.99	96.80%
Wikipedia hard	0.08	0.94	0.97	0.96	90.64%
Brown	0.03	0.99	0.97	0.98	93.94%

Character-based bi-LSTM model, Context window size is 15 trained with the early stopping method on the corresponding corpus

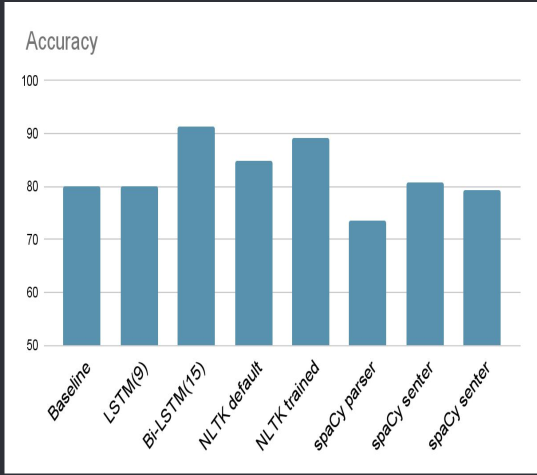


- Wikipedia hard has lower precision because in text set we do not split on semicolon, but in training set there are many such cases

Compare systems on the Wikipedia corpus

Character-based models, context for sentence ending punctuations, trained on 500,000 Wikipedia sentences with early stopping method, Evaluated on the Wikipedia test ground truth with hard cases.

Type	Error rate	Precision	Recall	F1-score	Accuracy
Baseline	0.16	0.88	0.93	0.91	80.12%
LSTM(9)	0.19	0.82	0.97	0.89	80.00%
Bi-LSTM(15)	**0.07**	**0.95**	**0.97**	**0.96**	**91.39%**
NLTK default	0.15	0.87	0.98	0.92	84.87%
NLTK trained, customized	0.09	0.94	0.97	0.95	89.07%
spaCy parser default	0.23	0.81	0.94	0.87	73.53%
spaCy senter default	0.16	0.90	0.92	0.91	80.84%
spaCy senter trained	0.18	0.88	0.92	0.90	79.44%



- Bi-LSTM has the best precision
- LSTM: low precision due to false positive predictions
- trained NLTK second best
- spaCy parser - dependency tree is broken by abbreviations, quotes

Test on hard cases and random sentences from the Wikipedia corpus

Context window	error rate	precision	recall	F1	Acc
Test set with hard cases (2928 sentences)					
bi-LSTM(17)					
for each basic unit	0.0737	0.9534	0.9703	0.9617	90.95%
for punctuations	0.0880	0.9349	0.9738	0.9540	90.37%
for sentence end marks	**0.0699**	**0.9537**	**0.9741**	**0.9638**	**91.36%**
LSTM(9)					
for sentence end marks	0.1707	0.8497	0.9719	0.9067	81.90%
Test set with random cases (3267 sentences)					
bi-LSTM(17)					
for each basic unit	0.0182	0.9867	0.9950	0.9908	98.03%
for punctuations	0.0222	0.9830	0.9947	0.9888	97.68%
for sentence end marks	**0.0145**	**0.9882**	**0.9972**	**0.9927**	**98.28%**
LSTM(9)					
for sentence end marks	0.0511	0.9561	0.9878	0.9717	93.88%

The bi-LSTM models are trained on 500,000 Wikipedia sentences for 10 epochs the LSTM model on 3,787,371 sentences for 10 epochs.

- All systems perform better on random sentences
- LSTM improved most

⇒ SBD highly syntax dependent

Training data size/2

Training sentences	token based full text	token based punct.	token based end marks	character based full text	character based punct.	character based end marks
			LSTM(9)			
7,000	75.41	75.52	75.48	78.59	77.52	78.50
500,000	80.02	78.28	79.30	81.11	81.04	81.18
3,780,371	**80.28**	**82.67**	**79.99**	**81.40**	**82.49**	**82.02**
			bi-LSTM(17)			
7,000	88.59	87.29	87.47	85.47	87.90	87.40
500,000	89.83	90.45	91.25	**91.90**	**91.97**	91.68
3,780,371	**89.90**	**91.32**	**91.54**	91.54	90.74	**91.97**

- Graph on the slide 22

Number of sentences	Context window created for every character: Memory	character: Window	token: Memory	token: Window	punctuation: Memory	punctuation: Window
7,000	28 MB	790k	10 MB	282k	1 MB	25k
500,000	1,25 GB	34,952k	472 MB	13,133k	48 MB	1,334k
3,780,371	9 GB	254,460k	3 GB	95,308k	336 MB	9,338k

- With different methods for creating the context window we optimize the memory usage and the training time

Rule-based approach

Find sentence boundaries with predefined rules:

Sentence end mark + whitespace + capital letter:

The sun is shining. The weather is sunny.

But: Mr. Brown is reading.

- Handcrafted rules, time and effort consuming
- Use a list of abbreviations
- Difficult to cover each case
- System is fragile due too many special cases
- Hard to maintain and customize

Statistical approach

Collect statistics about:

- occurrences of tokens, punctuations,
- token length,
- casing
- collocations of tokens

Calculate probabilities and test some assumptions:

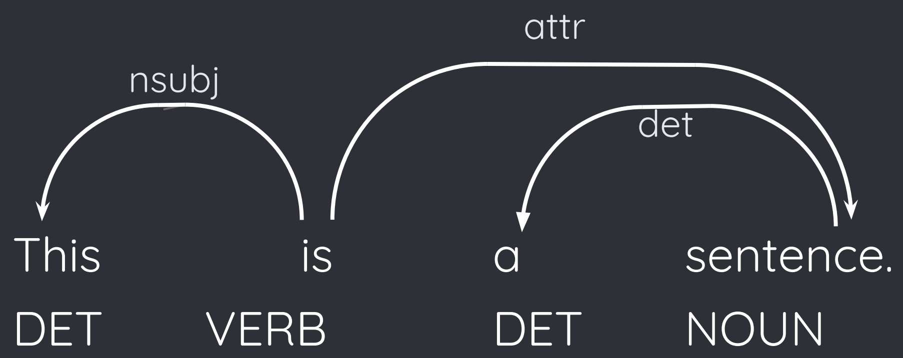
- frequent sentence starter
- token’s collocation with a period

Trainable with unsupervised training (NLTK)

- No annotation needed

Dependency tree

how the words within a sentence are related to each other



Embeddings

- Vector representation for words or characters
- capture meaningful information
- Learned during the training process

Input	Word to index	Embeddings
in	7	[0.342 -0.025 -1.690 0.717]
	3	[-0.643 2.726 0.074 0.696]
the	4	[1.497 1.344 -0.965 3.453]
	3	[-0.643 2.726 0.074 0.696]
U	52	[0.342 -0.025 -1.690 0.717]
.	6	[0.222 -3.025 -1.650 0.237]
S	34	[0.543 -0.021 -1.950 0.377]
.	6	[0.222 -3.025 -1.650 0.237]
A	8	[-0.112 -0.032 1.690 0.754]
.	6	[0.222 -3.025 -1.650 0.237]
	3	[-0.643 2.726 0.074 0.696]

Load Raw Data → Split Data → Create vocabulary → Encode vocabulary → Encode units → Create context windows

Load Raw Data

Input:

A cat has nine lives. A journey of thousand miles begins with a single step.

Output:

[‘A cat has nine lives.’, 'A journey of thousand miles begins with a single step.’]

Split Data

[[‘A’, ’ ‘, ’cat’, ’ ‘, ’has’, ’ ‘, ’nine’, ’ ‘, ’lives’, ’.’], [‘ ’, A’, ’ ‘, ’journey’, ’ ‘, ’of’, ’ ‘, ’thousand’, ’ ‘, ’miles’, ’ ‘, ’begins’, ’ ‘, ’with’, ’ ‘, ’a’, ’ ‘, ’single’, ’ ‘, ’step’, ’.’]]

Create vocabulary

unit	# occurrence
‘ ‘	3095
,	295
a	1563
of	512
begins	2
lives	63
……	
miles	1

Encode vocabulary

word	index
_PAD	0
_UNK	1
_DIG	2
‘ ‘	3
,	4
a	5
of	6
………	

Encode units

Input:

[[‘A’, ’ ‘, ’cat’, ’ ‘, ’has’, ’ ‘, ’nine’, ’ ‘, ’lives’, ’.’], [‘ ’, A’, ’ ‘, ’journey’, ’ ‘, ’of’, ’ ‘, …

Output:

[[5, 3, 9, 3, 13, 3, 16, 3, 23, 7], [5, 3, 1, 3, 6, 3, 11, 3, 1, 3, 14, 3, 6, 3, 5, 3, 12, 3, 19, 7]]

Create context windows

[[0, 0, 0, 0, 5]
[0, 0, 0, 5, 3]
[0, 0, 5, 3, 9]
[0, 5, 3, 9, 3]
[5, 3, 9, 3, 13]
[3, 9, 3, 13, 3]
[9, 3, 13, 3, 16]
[3, 13, 3, 16, 3]
[13, 3, 16, 3, 23]
[3, 16, 3, 23, 7]
……….]

Process of data preparation

F1-score

$$\text{F1-score} = \left(\frac{\text{Precision}^{-1} + \text{Recall}^{-1}}{2}\right)^{-1} = \frac{2}{\frac{(FP+TP) + (FN+TP)}{TP}} = \frac{2TP}{FP+TP + FN+TP} = \frac{TP}{TP+\frac{1}{2}(FP+FN)}$$

$$= \frac{2TP^2}{TP(FP+TP) + TP(FN+TP)}$$

$$= \frac{\frac{2TP^2}{(FP+TP)(FN+TP)}}{\frac{TP(FP+TP) + TP(FN+TP)}{(FP+TP)(FN+TP)}}$$

$$= 2\,\frac{\frac{TP}{(FP+TP)} \cdot \frac{TP}{(FN+TP)}}{\frac{TP(FP+TP) + TP(FN+TP)}{(FP+TP)(FN+TP)}}$$

$$= 2\,\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$F_{\square}$-score / 2

apply additional weight β to consider recall β-times more than precision

$$F_\beta\text{-score} = (1 + \beta^2)\,\frac{\text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

$\beta > 1$ weighs recall higher than precision

Threshold

Threshold $\in [0.01, 1[$

steps of 0.01

Calculate F1-score

Save the threshold belonging to the best F1-score

Training setup

- batch size of 256:
  number of context windows in one iteration of the training,
- embeddings size of 256:
  the length of the feature vector encoding a basic unit
- number of units 128: the LSTM layer includes this number of neurons (dimension of the hidden layer)
- number of layers 1
- learning rate 0.01: the size of the steps in adjusting the weights during the training in order to minimize loss
- stochastic gradient descent (SGD) optimizer algorithmus to minimize loss
- cross entropy loss function to measure the correctness of the prediction during the training

Early stopping

- Loss
  - difference between the labels and the model output
- Validation
  - Calculate loss after a certain number of epochs
- Early stopping
  - If the validation loss not decreasing, stop the training process
- Patience
  - Number of epochs we wait before we stop the training

Token-based context window

Input text:	‘The average temperature is 20.7 °C.’, ‘Most rain falls in the winter.’

	LSTM	bi-LSTM
For each basic unit	PAD PAD PAD PAD The PAD PAD PAD The	PAD PAD The average PAD The average
For punctuations	is 20 . 20 . 7 °	20 . 7 7 ° c .
For sentence end marks	is 20 . 20 . ° c .	20 . 7 ° c . Most

Character based context windows

	LSTM	bi-LSTM
For each basic unit	[PAD][PAD][PAD][PAD][T] [PAD][PAD][PAD][T][h]	[PAD][PAD][T][h][e] [PAD][T][h][e][ ]
for punctuations	[s][ ][2][0][.] [0][.][7][ ][°]	[2][0][.][7][ ] [7][ ][°][c][.]
for sentence end marks	[s][ ][2][0][.] [7][ ][°][c][.]	[2][0][.][7][ ] [°][c][.][ ][M]

Performance in English

Punkt from NLTK

- Error rate 1.02% on Brown and 1.65% on WSJ

spaCy parser

- F1-score 0.90 on OntoNotes 5.0

large corpus comprising various genres of text (news, conversational telephone speech, weblogs, usenet newsgroups, broadcast, talk shows)

Our baseline

	Error rate	F1-score
Brown	21%	0.88
Wall Street Journal	24%	0.86

Baseline algorithm evaluated on different corpora

Corpus	Error rate	Precision	Recall	F1-score	Accuracy
arXiv	0.08	0.94	0.97	0.96	92.30%
Europarl	0.07	0.97	0.95	0.96	90.98%
Wikipedia	0.16	0.88	0.93	0.91	80.12%
Brown	0.21	0.94	0.83	0.88	67.43%
WSJ	0.24	0.83	0.89	0.86	69.78%

- Strong lower bound on Europarl and arXiv
- formulas and variables at sentence end in arXiv do not cause errors
- abbreviations like *Senator J. W. Fulbright* lower precision
- titles without any punctuation cause missed sentence boundaries
- Highest error rate for Brown and WSJ

Context window types

uni-LSTM model(9)	Error rate	Precision	Recall	F1-score	Accuracy
Token based whole	**0.1780**	**0.8591**	0.9501	**0.9023**	80.02%
Token based on punctuation	0.1906	0.8509	0.9432	0.8947	78.28%
Token based on end marks	0.2192	0.7994	**0.9711**	0.8769	79.30%
Character based whole	0.1957	0.8283	0.9653	0.8916	81.11%
Character based on punctuation	0.1899	0.8333	0.9628	0.8951	81.04%
Character based on end marks	0.1904	0.8333	0.9660	0.8948	**81.18%**

bi-LSTM model(17)	Error rate	Precision	Recall	F1-score	Accuracy
Token based whole	0.0787	0.9511	0.9671	0.9590	89.83%
Token based on punctuation	0.0754	**0.9528**	0.9689	0.9608	90.45%
Token based on end marks	0.0740	0.9481	0.9755	0.9616	91.25%
Char. based whole	**0.0720**	0.9501	0.9755	0.9526	91.90%
Character based on punctuation	0.0733	0.9478	0.9766	**0.9620**	**91.97%**
Character based on end marks	0.0787	0.9388	**0.9802**	0.9590	91.68%

- the token- and character-based approaches for creating the input
- three different context window method
  - the character-based models overall better
  - context window for each unit inefficient for memory and time
  - context window for every punctuation mark reduces training data

Why we need sentences?

Used in other tasks within NLP

- POS-tagging

Book	*verb*
a	*determiners*
room.	*noun*

- Sentiment analysis:

It's great!	😁
We will be there in 5 Min.	😐
It was a bad game.	😞

- Text correction

How much is the temperature?

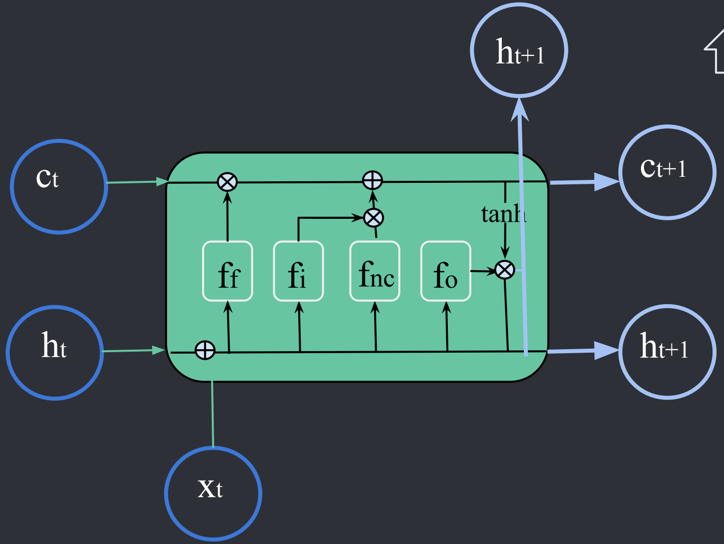
What is the temperature?

- Machine translation

🇩🇪 Pink ist meine Lieblingsfarbe.

🇬🇧 *Pink is my favourite color.*

Gates in LSTM



⊗ element wise multiplication

Forget gate:

- $f_f = \sigma(W_f(h_t, x) + b_f)$
- filters the old state c_t and decides what information to discard: $c_t \otimes f_f$

Input gate:

- $f_i = \sigma(W_i(h_t, x) + b_i)$
- decides what information to let through from the current input x

New candidate:

- $f_{nc} = \tanh(W_{nc}(h_t, x) + b_{nc})$
- Regulate the input between -1 and +1 with $\tanh(z) = \dfrac{e^z - e^{-z}}{e^z + e^{-z}}$

Gates in LSTM

Output gate:

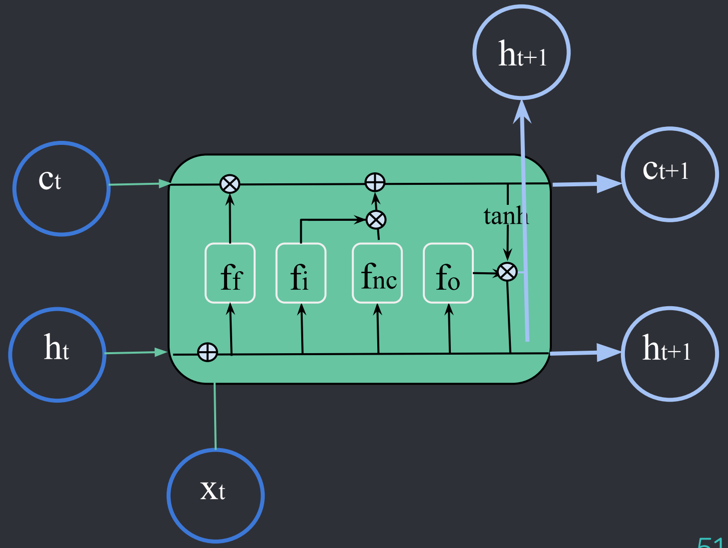
- $f_o = \sigma(W_o(h_t, x) + b_o)$
- calculates how much of the input is used in our output h_{t+1}

Cell state:

- $c_{t+1} = (c_t \otimes f_f) \oplus (f_i \otimes f_{nc})$
- a kind of memory chaining through all the time steps.

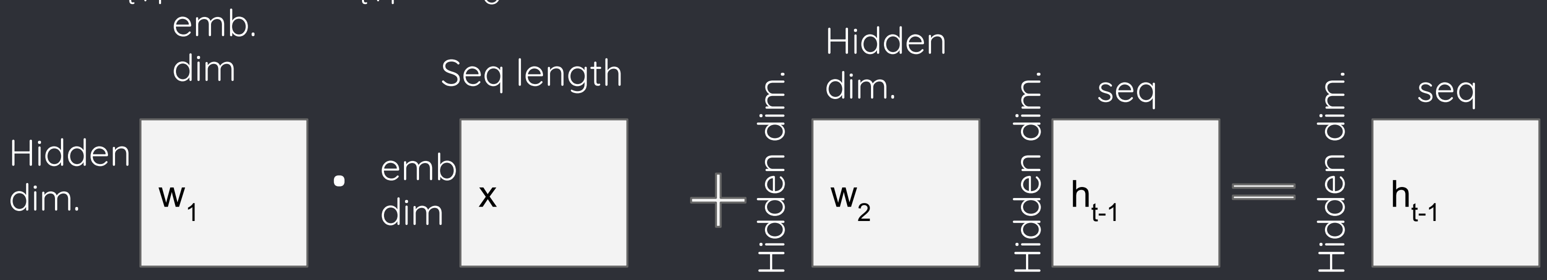
Hidden state:

- $h_{t+1} = \tanh(c_{t+1}) \otimes f_o$
- Output of the network



Dimensionality and LSTM computation

- $f_f = \sigma(W_f(h_t, x) + b_f)$
- $f_i = \sigma(W_i(h_t, x) + b_i)$
- $f_{nc} = \tanh(W_{nc}(h_t, x) + b_{nc})$
- $f_o = \sigma(W_o(h_t, x) + b_o)$
- $c_{t+1} = (c_t \otimes f_f) \oplus (f_i \otimes f_{nc})$
- $h_{t+1} = \tanh(c_{t+1}) \otimes f_o$



LSTM Model architecture