Finding Semantic Units in Deep Language Models

Bachelor's Thesis

Jasmin Denk

July 31th, 2018

Chair for Algorithms and Data Structures
Department of Computer Science
University of Freiburg

Sentiment analysis

Determine writer's attitude in a piece of text:

"I liked this book." $\rightarrow$ positive

"I didn't like this book." $\rightarrow$ negative

"While the characters were exceptionally well written, the story was very predictable." $\rightarrow$ neutral?

Motivation and approach

- Sentiment analysis: interesting for research and businesses
- Approach of Radford et al. (2017)[1]:
  - neural language model learns concept of sentiment by predicting next character in reviews
  - performs exceptionally well on multiple sentiment datasets
  - one unit seems to be responsible for results

[1] Radford, A., Jozefowicz, R., & Sutskever, I. (2017). Learning to generate reviews and discovering sentiment.

Goals of this thesis

- Reproduce results of Radford et al.
- Analyse how size of training data influences the results
- Transfer this approach to other semantic classification problems

Contents

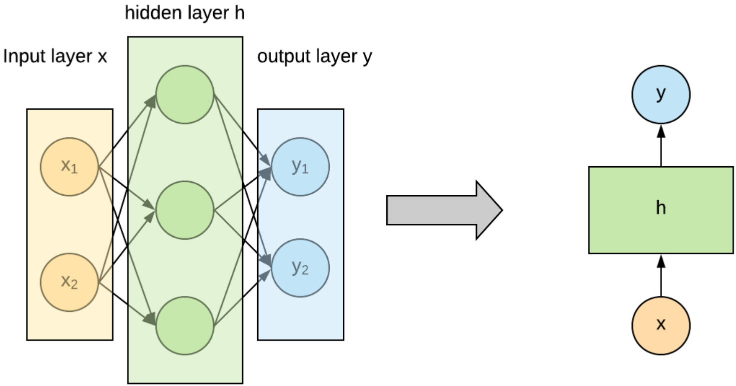
Preliminaries

Language models (character-level)

- Probability distribution over a sequence of characters
- Use to make probabilistic predictions:
    - $\mathrm{P}(c_n | c_{n-t-1} c_{n-t} \ldots c_{n-1})$ for character c_n dependent on t previous ones
    - E.g. $\mathrm{P}(o|hell) > \mathrm{P}(q|hell)$
- Neural language model: language model based on neural networks

Feed-forward neural networks (FFNN)

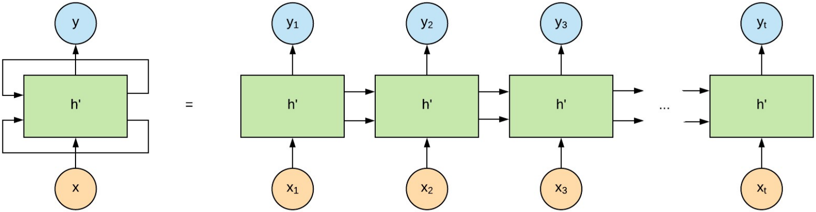
A FFNN consists of units which calculate an activation value and pass it to the next layer.



A feed-forward neural network with 2 input, 3 hidden and 2 output units (explicit and abbreviated visualization).

Recurrent NN with long short-term memory cells (LSTM)

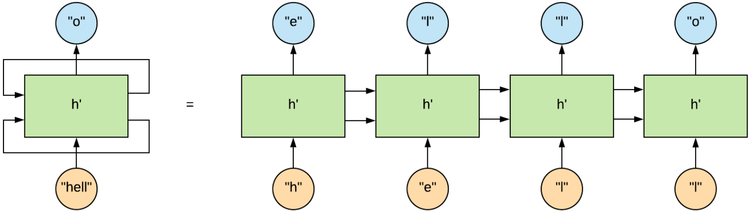
A LSTM consists of units which calculate an activation value, save and regulate it and pass it to the next layer; introducing the passed *cell state*, the passed *hidden state*, and multiple gates wrapping the hidden units.



A LSTM network (abbreviated and unrolled visualization).

Recurrent NN with long-short term memory cells (LSTM)

A LSTM consists of units which calculate an activation value, save and regulate it and pass it to the next layer; introducing the passed *hidden state*, the passed *cell state* and multiple gates wrapping the hidden units.



A trained LSTM network predicting "o" as next character given "hell" (abbreviated and unrolled visualization).

Basic approach

- Train LSTM to predict the next character of continuous text
- Cell state has to characterize input text for optimal prediction
  $\Rightarrow$ when text contains prominent features (e.g. sentiment), the cell state should learn to represent them
- Use cell state for classification

The developed system

Component: Language model

- Implement own LSTM language model using TensorFlow
- Important hyperparameters:
  - *num_units*: number of units in hidden layer
  - *seq_length*: max. number of characters directly influencing prediction
- Provide function: return final cell state of LSTM (vector consisting of *num_units* values) given text

Component: Classifier

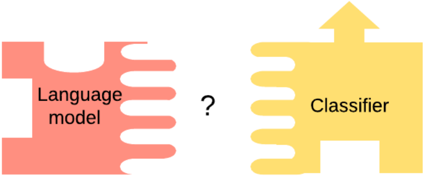
- Classify objects by given features
- Binary classification: label 1 (positive example), label 0 (negative example)
- Prediction of feature vector x based on decision function d(x):

$$d(x) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} w_i * x_i + b > 0, n\text{: number of features} \\ 0 & \text{else} \end{cases}$$

- Use cell state of language model given text as feature vector

What now?

How do these components interact?



Goal: Find semantic units in language model using classifier

Process of finding a semantic unit

1. Choose semantic concept and datasets

Search for matching datasets with similar semantic characteristics (e.g. reviews, e-mails, lyrics):

- For language model:
  - (plenty of) unlabelled text data

- For classifier:
  - labelled text data
  - divided in train/validation/test split

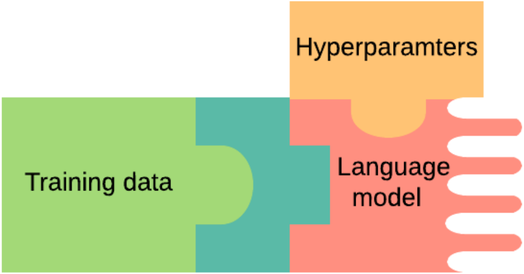
2. Pre-process data and 3. Choose hyperparameters

- Data consists of multiple instances
- Pad text instances with start token ("\n ") and end token (" ")
- Replace newlines with whitespaces, delete trailing whitespaces

- Select values of hyperparameters for language model

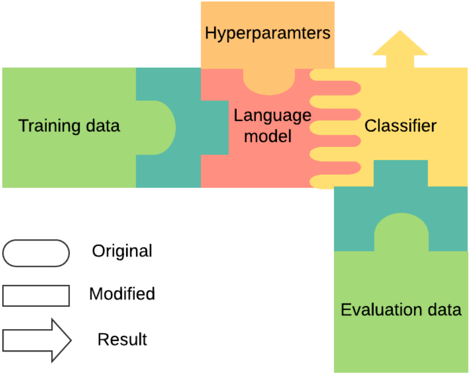
4. Train language model

- Train language model with chosen hyperparameters and pre-processed training data
- Trained model can be used to return final cell state when provided with text



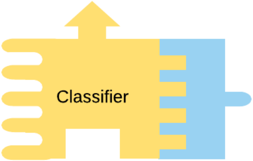
5. Train and evaluate classifier (using all hidden units)

- Let trained language model transform pre-processed evaluation data
- Train classifier given those *num_units* features
- Document evaluation result



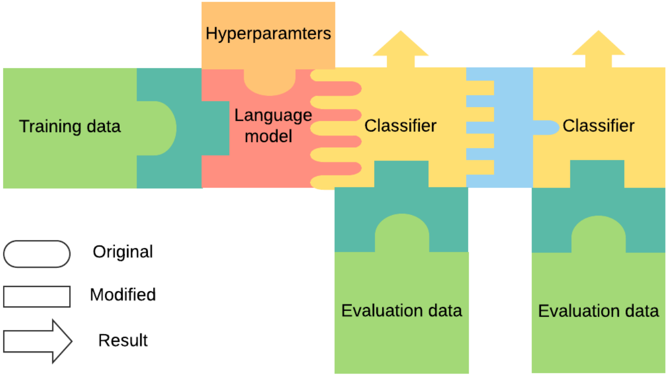
6. Analyse trained classifier

- Inspect weights of trained classifier associated with features
- Highest absolute weight $\rightarrow$ associated feature most relevant for correct classification
- Return feature with highest absolute weight $\hat{=}$ crucial hidden unit of language model ("semantic unit")



7. Train and evaluate classifier (using only semantic unit)

- Get activation value of crucial unit given pre-processed evaluation data
- Train classifier again given only this one feature
- Document evaluation result



What now?

- Not given that language model evolved crucial unit for semantic characteristic / concept
- Analyse results
  - different hyperparameters
  - different size of training data
  - different semantic classification tasks

Evaluation

Evaluation

- 3 different binary classification tasks:
  - Sentiment analysis ("positive" / "negative" review)
  - Spam classification ("spam" / "ham" email)
  - Mood classification ("happy" / "sad" lyric)
- Evaluation metrics:
  - accuracy: $\frac{\text{\# correctly classified examples}}{\text{\# examples}}$
  - recall: $\frac{\text{\# correctly classified positive examples}}{\text{\# actually positive examples}}$
  - precision: $\frac{\text{\# correctly classified positive examples}}{\text{\# as positive classified examples}}$
  - f1-score: $2 * \frac{\text{precision * recall}}{\text{precision} + \text{recall}}$

Sentiment analysis: Datasets

Choose training and evaluation data according to Radford et al. :

- For language model:
  - Amazon product review dataset[2]; more than 82 million reviews
  - 3 different sized subsets: 0.2 million, 2 million and 20 million reviews
- For classifier:
  - binary version of Stanford Sentiment Treebank[3]
  - train/validation/test split: 6920/872/1821 reviews

[2] He, R. and McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering.

[3] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank.

Sentiment analysis: Dataset excerpt

sentence	label
In its ragged, cheap and unassuming way, the movie works.	1
While the film misfires at every level, the biggest downside is the paucity of laughter in what's supposed to be a comedy.	0
I love the way that it took chances and really asks you to take these great leaps of faith and pays off.	1
Lacks heart, depth and, most of all, purpose.	0

Excerpt from the testing split of the binary SST dataset.

Sentiment analysis: Results

hyperparameters		all units used				only sentiment unit used				
seq_length	num_units	f1	recall	pred	acc	f1	recall	pred	acc	time
Language model trained on 0.2 million reviews										
100	1024	0.65	0.68	0.62	0.63	0.64	0.87	0.50	0.51	0.2h
	2048	0.67	0.71	0.66	0.65	0.64	0.82	0.52	0.53	0.7h
	4096	0.69	0.72	0.67	0.68	0.66	0.94	0.51	0.51	2.7h
200	1024	0.64	0.68	0.60	0.64	0.67	1.00	0.50	0.50	0.2h
	2048	0.64	0.69	0.59	0.61	0.67	1.00	0.50	0.50	0.6h
	4096	0.67	0.70	0.64	0.65	0.67	1.00	0.50	0.50	2.7h
Language model trained on 2 million reviews										
100	1024	0.75	0.76	0.73	0.74	0.71	0.80	0.65	0.68	2h
	2048	0.79	0.80	0.78	0.79	0.77	0.81	0.74	0.77	8h
	4096	0.83	0.84	0.82	0.83	0.81	0.84	0.79	0.81	32h
200	1024	0.71	0.74	0.69	0.70	0.66	0.65	0.66	0.66	3h
	2048	0.79	0.80	0.78	0.79	0.79	0.82	0.76	0.78	8h
	4096	0.82	0.83	0.80	0.81	0.79	0.83	0.75	0.78	33h
Language model trained on 20 million reviews										
100	1024	0.75	0.76	0.75	0.76	0.73	0.75	0.71	0.73	25h
	2048	0.85	0.88	0.81	0.84	0.84	0.89	0.89	0.84	79h
	4096	0.87	0.90	0.85	0.87	0.87	0.90	0.83	0.86	329h
200	1024	0.77	0.79	0.76	0.77	0.77	0.77	0.76	0.77	24h
	2048	0.85	0.86	0.84	0.84	0.79	0.83	0.76	0.78	80h
	4096	0.87	0.90	0.85	0.87	0.82	0.88	0.77	0.81	326h

Performances of the classifier on the binary SST dataset.
The respective language model was trained for 1 epoch.

Observations:

- no sentiment unit evolved using 0.2 million reviews
- the more training data, the better
- *seq_legth* mostly irrelevant
- *num_units*: 4096 > 2048 > 1024

Sentiment analysis: Results

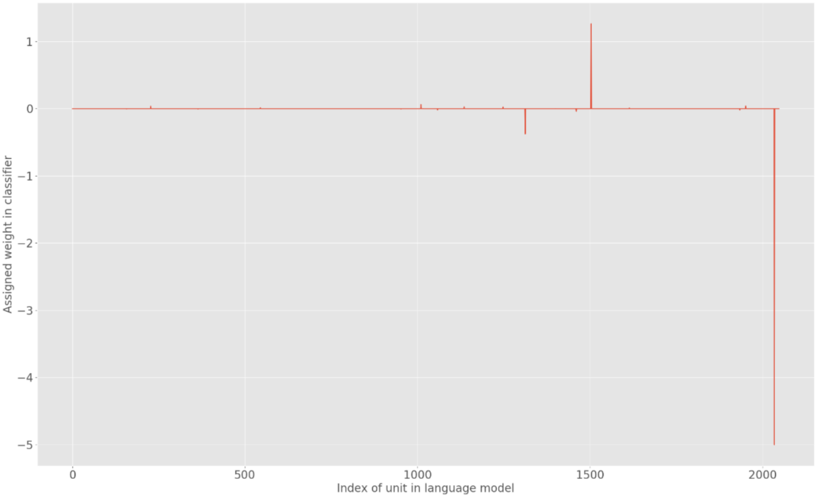
hyperparameters		all units used				only sentiment unit used				
seq_length	num_units	f1	recall	pred	acc	f1	recall	pred	acc	time
Language model trained on 0.2 million reviews										
100	1024	0.65	0.68	0.62	0.63	0.64	0.87	0.50	0.51	0.2h
	2048	0.67	0.71	0.66	0.65	0.64	0.82	0.52	0.53	0.7h
	4096	0.69	0.72	0.67	0.68	0.66	0.94	0.51	0.51	2.7h
200	1024	0.64	0.68	0.60	0.64	0.67	1.00	0.50	0.50	0.2h
	2048	0.64	0.69	0.59	0.61	0.67	1.00	0.50	0.50	0.6h
	4096	0.67	0.70	0.64	0.65	0.67	1.00	0.50	0.50	2.7h
Language model trained on 2 million reviews										
100	1024	0.75	0.76	0.73	0.74	0.71	0.80	0.65	0.68	2h
	2048	0.79	0.80	0.78	0.79	0.77	0.81	0.74	0.77	8h
	4096	0.83	0.84	0.82	0.83	0.81	0.84	0.79	0.81	32h
200	1024	0.71	0.74	0.69	0.70	0.66	0.65	0.66	0.66	3h
	2048	0.79	0.80	0.78	0.79	0.79	0.82	0.76	0.78	8h
	4096	0.82	0.83	0.80	0.81	0.79	0.83	0.75	0.78	33h
Language model trained on 20 million reviews										
100	1024	0.75	0.76	0.75	0.76	0.73	0.75	0.71	0.73	25h
	2048	0.85	0.88	0.81	0.84	0.84	0.89	0.89	0.84	79h
	4096	0.87	0.90	0.85	0.87	0.87	0.90	0.83	0.86	329h
200	1024	0.77	0.79	0.76	0.77	0.77	0.77	0.76	0.77	24h
	2048	0.85	0.86	0.84	0.84	0.79	0.83	0.76	0.78	80h
	4096	0.87	0.90	0.85	0.87	0.82	0.88	0.77	0.81	326h

Performances of the classifier on the binary SST dataset.
The respective language model was trained for 1 epoch.

Comparison to Radford et al.:

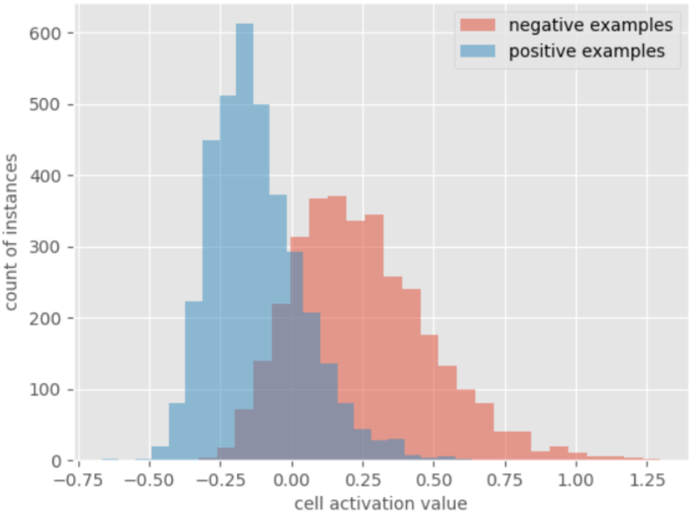
- 82 million reviews, *num_units*: 4096, *seq_legth*: 256
- using all hidden units: 91.8% accuracy
- using sentiment unit: not specified

Sentiment analysis: Visualisation



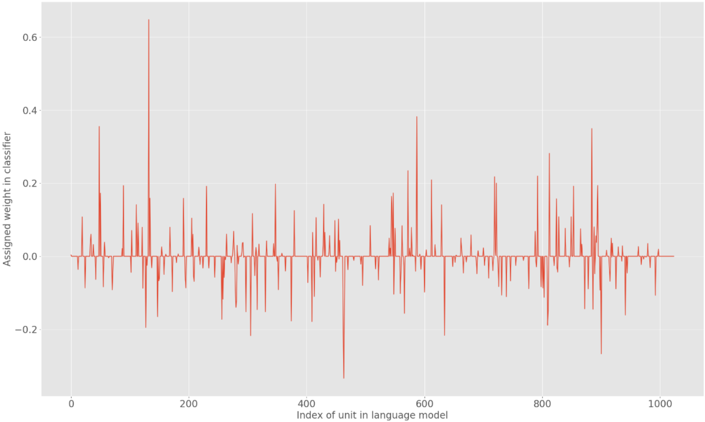
Graph representing the unit contributions of a classifier trained on the binary SST dataset.
The associated language model was trained on 20 million reviews with *num_units* 2048 and *seq_length* 100.

Sentiment analysis: Visualisation



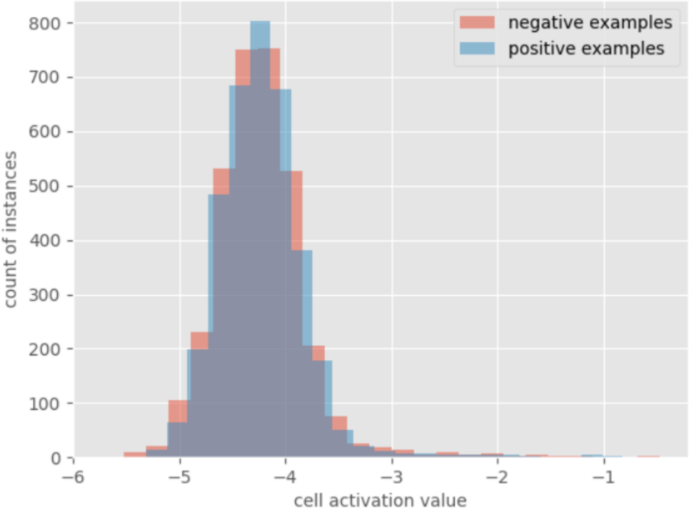
Histogram representing the cell activation values for the found sentiment unit (index 2034) on the training split of the binary SST dataset.

Sentiment analysis: Visualisation



Graph representing the unit contributions of a classifier trained on the binary SST dataset.
The associated language model was trained on 0.2 million reviews with *num_units* 1024 and *seq_length* 200.

Sentiment analysis: Visualisation



Histogram representing the cell activation values for the found sentiment unit (index 132) on the training split of the binary SST dataset.

Spam classification: Datasets

Create own training and evaluation dataset based on Enron Spam dataset[4]:

- For language model:
  - 23,220 emails
    - 8,175 spam emails
    - 15,045 ham emails
- For classifier:
  - train/validation/test split: 2100/300/600 emails
  - 1:1 spam-ham-ratio respectively

[4] Metsis, V., Androutsopoulos, I., and Paliouras, G. (2006). Spam filtering with naive bayes - which naive bayes?

Spam classification: Dataset excerpt

email	spam
Subject: young wifes click here to be removed	1
Subject: chart info here it is .	0
Subject: why pay for over priced pre \ scription dru @ gs ? ? ?	1
Subject: fw : revised michelle, sempra called on 21, 500 of needles space from 11 / 01 through 10 / 02 . please see attached memo from stepahie . thanks , tk [...]	0

Excerpt from the pre-processed testing split of the created spam dataset.

Spam classification: Results

hyperparameters		all units used				only spam unit used				
seq_length	num_units	f1	recall	pred	acc	f1	recall	pred	acc	time
100	1024	0.90	0.91	0.89	0.90	0.68	0.67	0.69	0.68	0.9h
	2048	0.93	0.96	0.91	0.93	0.63	0.71	0.56	0.58	2.1h
	4096	0.95	0.97	0.94	0.95	0.67	1.0	0.5	0.5	6.9h
200	1024	0.90	0.91	0.90	0.90	0.80	0.77	0.82	0.80	0.7h
	2048	0.93	0.95	0.91	0.93	0.66	0.70	0.63	0.64	1.9h
	4096	0.95	0.95	0.94	0.95	0.61	0.64	0.58	0.59	6.9h

Performances of the classifier on the created spam dataset.
The respective language model was trained for 5 epochs.

Observations:

- results using spam unit very lopsided
- *seq_length* (concerning spam units): 200 > 100
- *num_units* (concerning spam units): 1024 > 2048 > 4096

Spam classification: Results

hyperparameters		all units used				only spam unit used				
seq_length	num_units	f1	recall	pred	acc	f1	recall	pred	acc	time
100	1024	0.90	0.91	0.89	0.90	0.68	0.67	0.69	0.68	0.9h
	2048	0.93	0.96	0.91	0.93	0.63	0.71	0.56	0.58	2.1h
	4096	0.95	0.97	0.94	0.95	0.67	1.0	0.5	0.5	6.9h
200	1024	0.90	0.91	0.90	0.90	0.80	0.77	0.82	0.80	0.7h
	2048	0.93	0.95	0.91	0.93	0.66	0.70	0.63	0.64	1.9h
	4096	0.95	0.95	0.94	0.95	0.61	0.64	0.58	0.59	6.9h

Performances of the classifier on the created spam dataset.
The respective language model was trained for 5 epochs.

Comparison to baseline algorithm:

f1	recall	precision	accuracy	time
0.92	0.98	0.88	0.92	1s

Performances of the baseline algorithm on the created spam dataset.

Mood classification: Datasets

Analyse if a given lyric is "happy" or "sad":

- Change during pre-processing: Replace newlines with "#"
- For language model:
  - songdata dataset[5]
  - 57,650 lyrics
- For classifier:
  - MusicMood dataset[6]
  - train/validation/test split: 900/100/200 lyrics
  - labels manually assigned

[5] https://www.kaggle.com/mousehead/songlyrics (18.06.2018)

[6] Raschka, S. (2016). MusicMood: Predicting the mood of music from song lyrics using machine learning.

Mood classification: Dataset excerpt

lyric	mood
Where, oh, where have you been, my love?#Where, oh, where can you be?#It's been so long since the moon has gone#And, oh, what a wreck you've made me## [...]	0
This kind of love makes me feel ten feet tall#It makes all my problems fall#And this kind of trust helps me to hold the line#I'll be there every time## [...]	1
I'm a pop star threat and I'm not dead yet#Got a super-dread-bet with an angel drug-head#Like a dead beat winner, I want to be a sinner#An idolized bang for the industry killer## [...]	0
Country day#A day in the unknown#A gentle breeze gently blowing#Country day#Country day#Another day in the unknown#I can feel it in my bones#Country day## [...]	1

Excerpt from the pre-processed testing split of the MusicMood dataset.
The respective language model was trained for 5 epochs.

Mood classification: Results

hyperparameters		all units used				only mood unit used				
seq_length	num_units	f1	recall	pred	acc	f1	recall	pred	acc	time
100	1024	0.32	0.21	0.69	0.54	0.33	0.22	0.64	0.53	2.2h
	2048	0.58	0.51	0.68	0.62	0.22	0.13	0.67	0.51	4.2h
	4096	0.60	0.50	0.73	0.64	0.40	0.29	0.68	0.56	12.7h
200	1024	0.43	0.32	0.64	0.55	0.07	0.04	0.67	0.49	1.6h
	2048	0.57	0.51	0.65	0.60	0.0	0.0	0.0	0.48	3.5h
	4096	0.60	0.56	0.63	0.60	0.0	0.0	0.0	0.48	12.1h

Performances of the classifier on the MusicMood dataset.
The respective language model was trained for 5 epochs.

Observations and comparison:

- overall results underwhelming
- no evolved mood units
- Raschka: 72.5 % accuracy

Conclusions

- Own system approximates result of Radford et al. (87% vs. 92% using all units)
- System finds evolved semantic unit in language model if it exists
- Approach applicable to different semantic classification tasks with mixed results
- Size of training data very important

Demo

Thank you for your attention.

Appendix: Other used implementation

While developing the language model, we used another implementation[7] for preliminary results:

- Sentiment analysis:
  - similar results with small fluctuations
- Spam classification:
  - *num_units* 1024, *seq_length* 100
  - all units: 91%
  - spam unit: 87%
- Mood classification:
  - *num_units* 1024, *seq_length* 100
  - all units: 62%
  - mood unit: 61%

[7] https://github.com/crazydonkey200/tensorow-char-rnn (21.5.2018)

Appendix: Composition of sentiment training data

	composition of subset		
size of subset	positive reviews	negative reviews	neutral reviews
200,000	100,000	100,000	0
2,000,000	1,000,000	1,000,000	0
20,000,000	15,643,930	2,654,532	1,701,538

Composition of the used Amazon product data subsets. We call a review positive if the respective star-rating is 4 or 5, negative if it is 1 or 2 and neutral if is is 3.

Appendix: Analyse Complexity

Complexity for training language model[8] :

- Computational complexity:
  - $O(num\_units^2)$
  - observe: when doubling *num_units*, runtime quadruples
  - $O((num\_units * 2)^2) = O(4 * num\_units^2)$
- Space complexity:
  - $O(num\_units^2)$
  - observe: when doubling *num_units*, size of savefiles quadruple
  - but: After reading whole training data, it stays in memory $\Rightarrow$ $O(num\_units^2 + s)$; s: size of training data

[8] Gers, F. A. (2001). Long short-term memory in recurrent neural networks.

Appendix: Analyse Complexity

Complexity for training and evaluating classifier:

- linear regarding number of examples of respective split
- doubling *num_units* 1024 → 2048: runtime roughly doubles
- doubling *num_units* 2048 → 4096: runtime roughly quadruples

	time spend on feature vector creation			
num_units	training split	validation split	testing split	total
1024	20min	4min	7 min	33 min
2048	50min	8min	16min	75 min
4096	235min	40min	81min	360min

Average running time of the sentiment classifier. In the used binary SST dataset, the training split consists of 6920, the validation split of 872 and the testing split of 1821 examples. The classifiers were trained on a PC with Intel(R)Core(TM)i7-6700HQ CPU @ 2.60GHz processor and 16GB RAM.

Appendix: Text generation

starting text	predicted continuation
I hated this book! It was	a waste of time and money. so boring and the characters were so bad that I couldn't even finish it. a little difficult to read and the story was so bad that I was really disappointed in the story.
I loved this book! It was	a great read and I would recommend it to anyone who likes a good romance. a great read and I couldn't put it down. a great read and I was so excited to read the next book in the series.

Example text generation from different language models given two starting texts. The predicted next character was treated as the actual next character to let the language models continue the sentence.

Appendix: Neural language models[9]

- Language model based on neural networks
- Use ability of neural networks to learn distributed representations
  - vector of features characterizing the meaning of given text
  - learning algorithm should discover these features
  - idea: sentiment can be such a feature
- Different types of neural networks

[9] Yoshua Bengio (2008) Neural net language models. Scholarpedia, 3(1):3881.

Appendix: Language models detailed

"A language model is a function, or an algorithm for learning such a function, that captures the salient statistical characteristics of the distribution of sequences of words in a natural language, typically allowing one to make probabilistic predictions of the next word given preceding ones."[10]

- Here: Character level language models
- E.g. $\mathrm{P}(e|\mathit{positiv}) > \mathrm{P}(q|\mathit{positiv})$
- Neural language model: Language model based on neural networks

[10] Yoshua Bengio (2008) Neural net language models. Scholarpedia, 3(1):3881.

Appendix: Sentiment analysis detailed

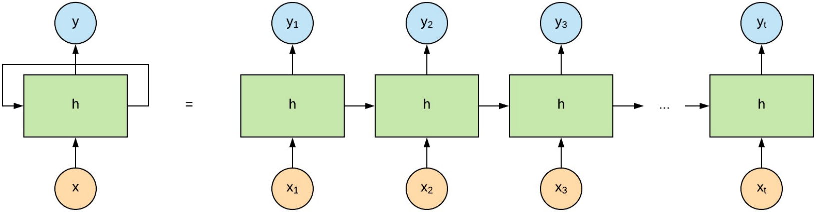
"The process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral."[11]

"I liked this book." $\rightarrow$ positive
"I didn't like this book." $\rightarrow$ negative
"While the characters were exceptionally well written, the story was very predictable." $\rightarrow$ neutral?

[11] Oxford dictionary

Appendix: Recurrent neural networks (RNN)

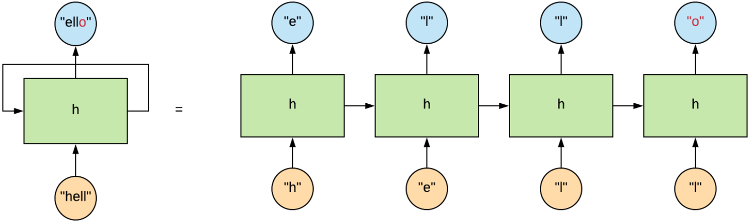
A RNN consists of units which calculate an activation value, save it and pass it to the next layer; introducing the passed *hidden state*.



A recurrent neural network (abbreviated and unrolled visualization).

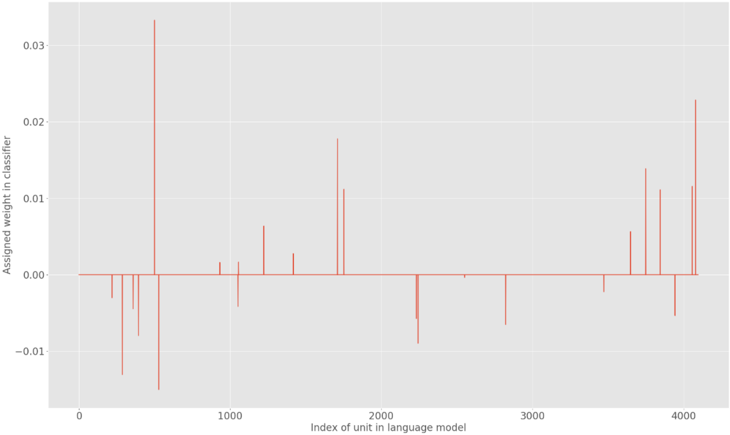
Appendix: Recurrent neural networks (RNN)

A RNN consists of units which calculate an activation value, save it and pass it to the next layer; introducing the passed *hidden state*.



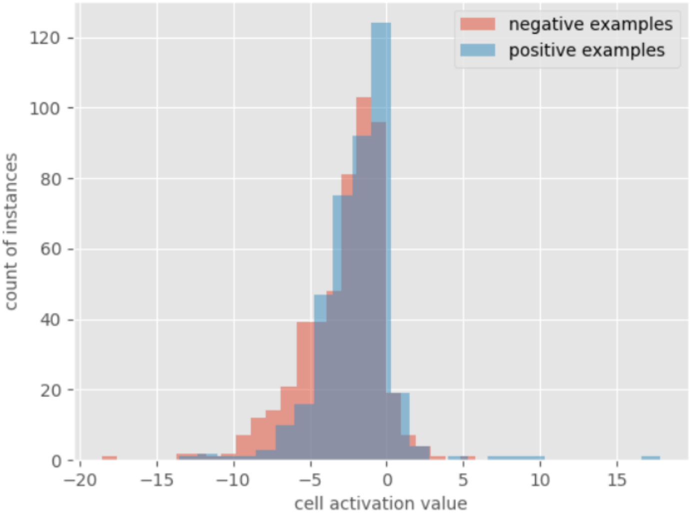
A recurrent neural network (abbreviated and unrolled visualization).

Appendix: Mood classification: Visualizations



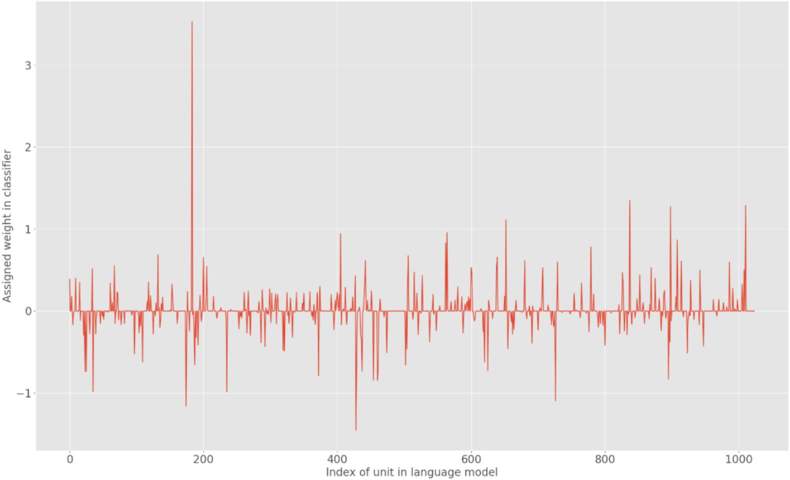
Graph representing the unit contributions of a classifier trained on the MusicMood dataset.
The associated language model was trained with *num_units* 4096 and *seq_length* 100.

Appendix: Mood classification: Visualizations



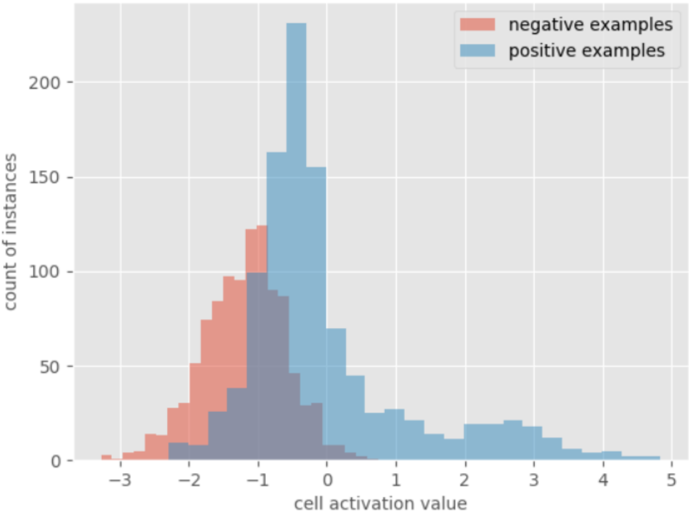
Histogram representing the cell activation values for the found mood unit (index 500) on the training split of the MusicMood dataset.

Appendix: Spam classification: Visualizations



Graph representing the unit contributions of a classifier trained on the created spam dataset. The associated language model was trained with *num_units* 1024 and *seq_length* 200.

Appendix: Spam classification: Visualizations



Histogram representing the cell activation values for the found spam unit (index 183) on the training split of the created spam dataset.