Sentiment Analysis on Movie Reviews

Ahmed Sabir, Alexandra Yamaui, Jakob Gerstenlauer, Juan Carlos Largo, Martin Isaksson

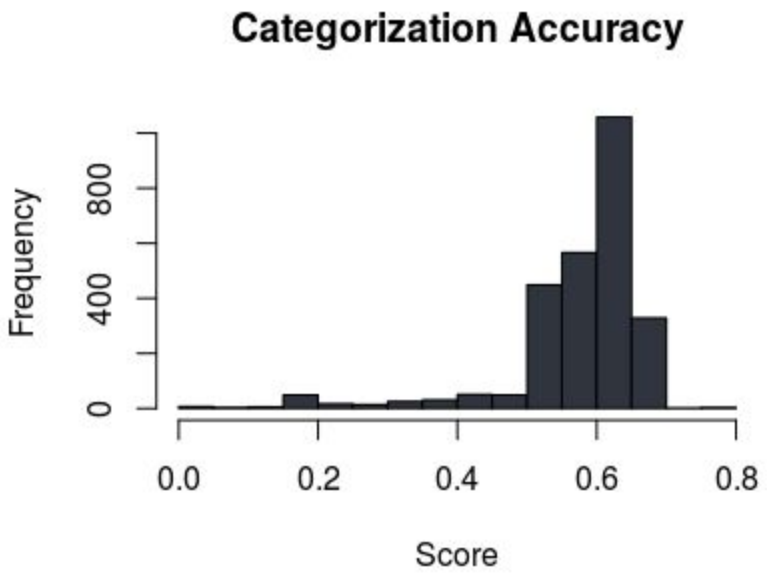
2017-01-31

Introduction

kaggle Competition

Dataset:

- Phrases from the Rotten Tomatoes dataset.
  - Train set:
    - Phrases and their associated sentiment labels
  - Test set:
    - Just phrases



Max Score: 0.76527

Introduction

kaggle Competition

Dataset:

- **Sentences**
  - Half of which considered positive
  - and the other half, negative
- **Phrases**
  - Every syntactically plausible combination of the sentences
  - According to Socher et al. the classification works better with shorter phrases.

Introduction

kaggle Competition

Goal:

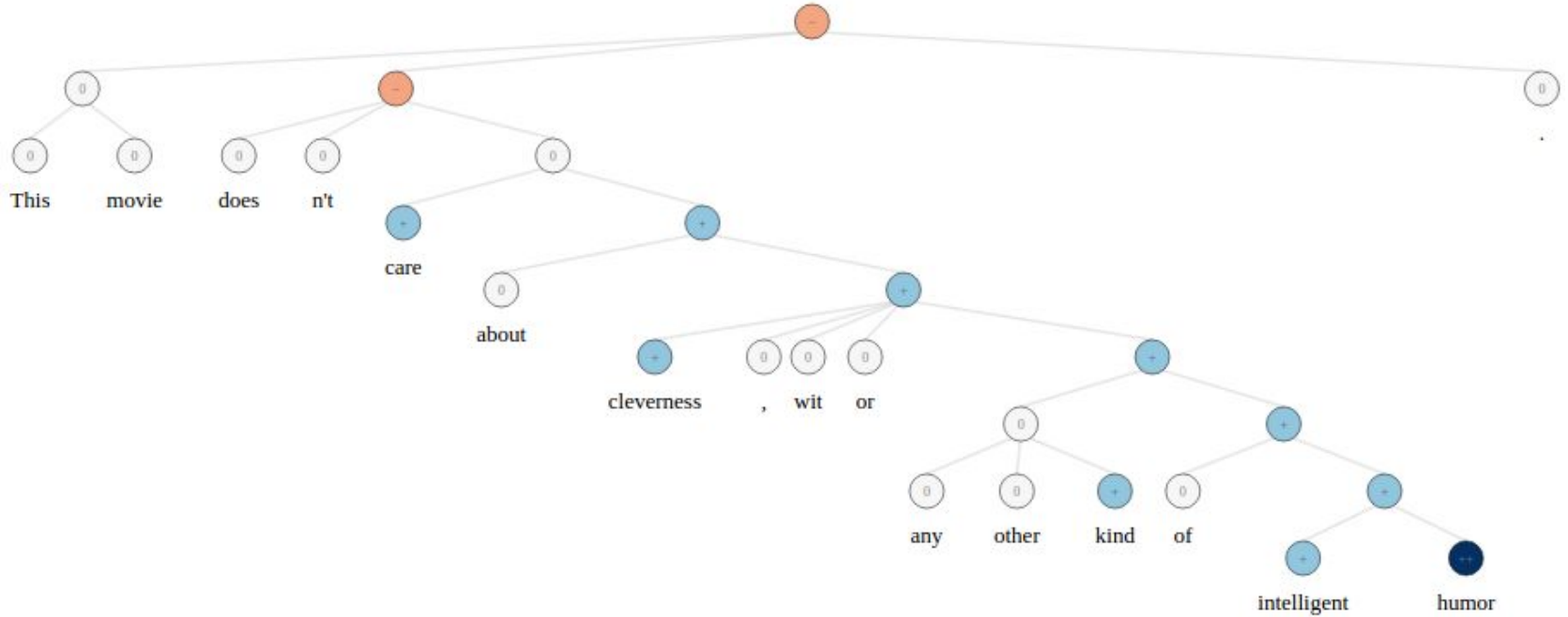
- Label phrases on a scale of five values
  - **0 - negative**
  - **1 - somewhat negative**
  - **2 - neutral**
  - **3 - somewhat positive**
  - **4 - positive**

Obstacles:

- Sentence negation
- Sarcasm
- Terseness
- Language ambiguity

Introduction

“This movie doesn't care about cleverness, wit or any other kind of intelligent humor.”



Datasets

- Dataset size: 222352 sentences, phrases and single tokens (train: 156060 , test: 66292)
- Train data: 52.7% (75%)
- Validation data: 17.5% (25% of train data)
- Test data: 29.8%
- Dictionary: 13759 different tokens
- Sentences: 8544 (train), 3311 (test)

PhraseId	SentenceId	Phrase	Sentiment
1	1	A series of escapades demonstrating	
2	1	A series of escapades demonstrating	
3	1	A series	2
4	1	A	2
5	1	series	2
6	1	of escapades demonstrating the adage	
7	1	of	2
8	1	escapades demonstrating the adage th	
9	1	escapades	2
10	1	demonstrating the adage that what is	
11	1	demonstrating the adage	2

Pre-Processing the Datasets

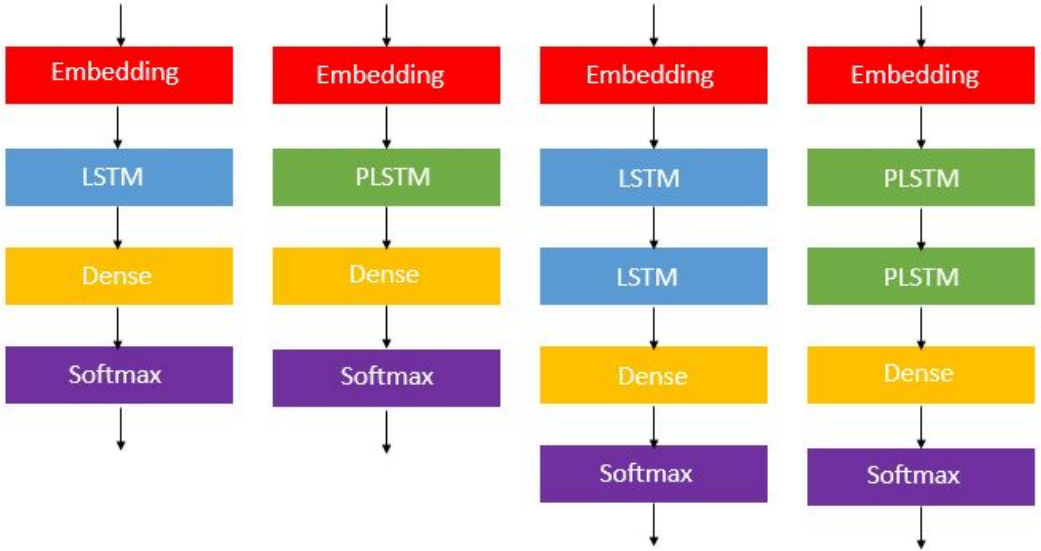
- Tokenizer
- Remove stopwords
- SnowballStemmer
- Create token-id
- Connect id with correct label
- One-hot encoding for labels
- Zero-padding for data (size of pre-processed data: 192660)
- Single tokens are classified with label 2

```
array([[ 5, 11,  6,  7, 11, 10,  1, 12,  3,  9,  0, 14],
       [ 0,  0,  0,  0,  0, 13,  4,  2,  8,  5, 11,  6],
       [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 13,  4],
       [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 13],
       [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  4]])
```

```
array([[ 0.,  1.,  0.,  0.,  0.],
       [ 0.,  0.,  1.,  0.,  0.],
       [ 0.,  0.,  1.,  0.,  0.],
       [ 0.,  0.,  1.,  0.,  0.],
       [ 0.,  0.,  1.,  0.,  0.]])
```

Models & Hyperparameters

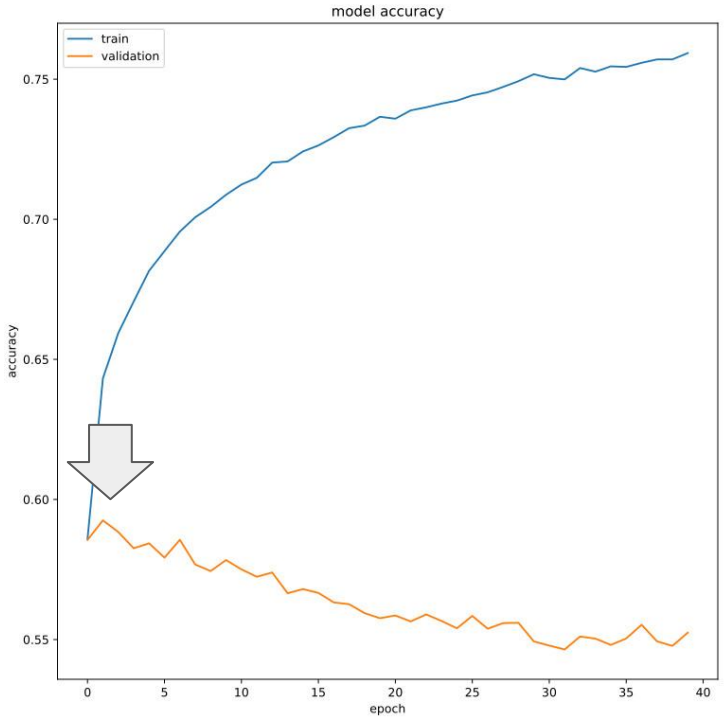
- Units in Embedding-/LSTM-/PLSTM-layer: 128
- Units in Dense-layer: 5
- Drop-out rates
  - Two left models: 0.2, 0.3, 0.6, 0.8
  - Two right models:
  - 0, 0.2
- Drop-out in Embedding-/LSTM-layers
- Optimizers: Adam & RMSProp (respectively left to right)
- Batch-size: 128
- Learning-rate: 0.001
- Epochs: 40
- Loss: Categorical crossentropy



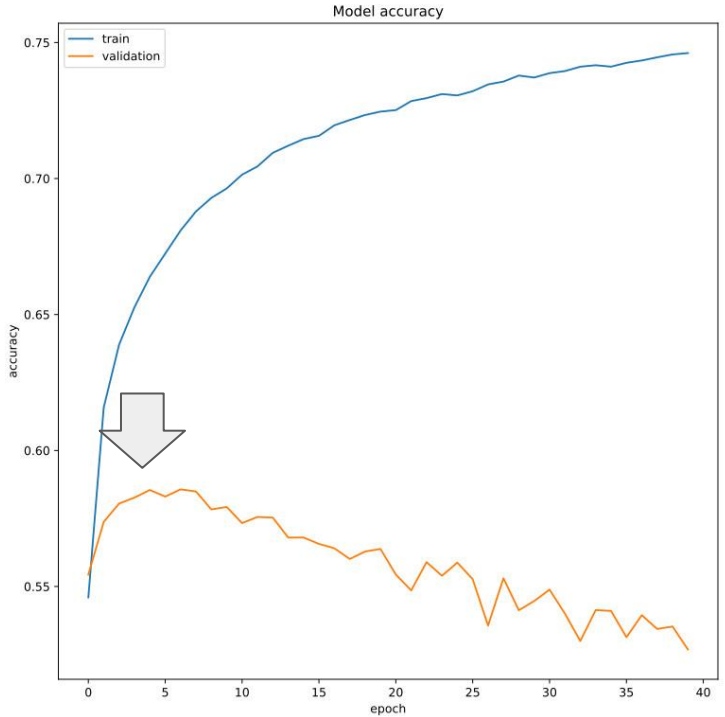
Results

using a 20% drop-out rate

LSTM

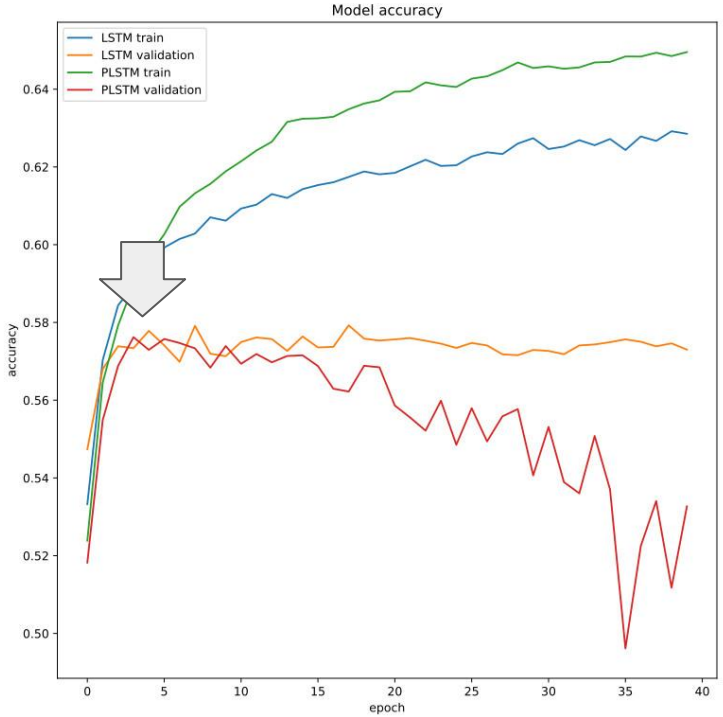


PLSTM

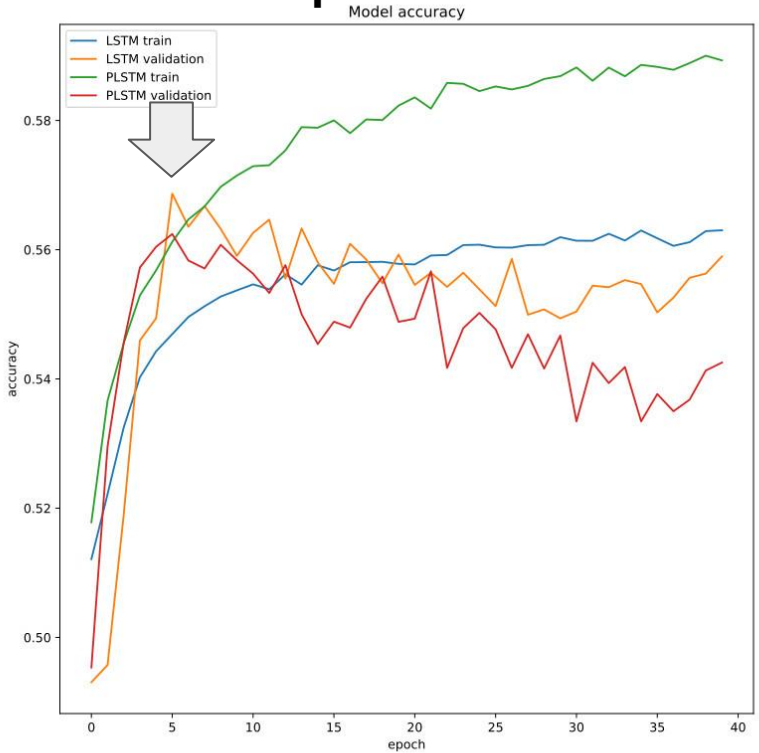


Increasing the drop-out rate...

60% drop-out rate



80% drop-out rate

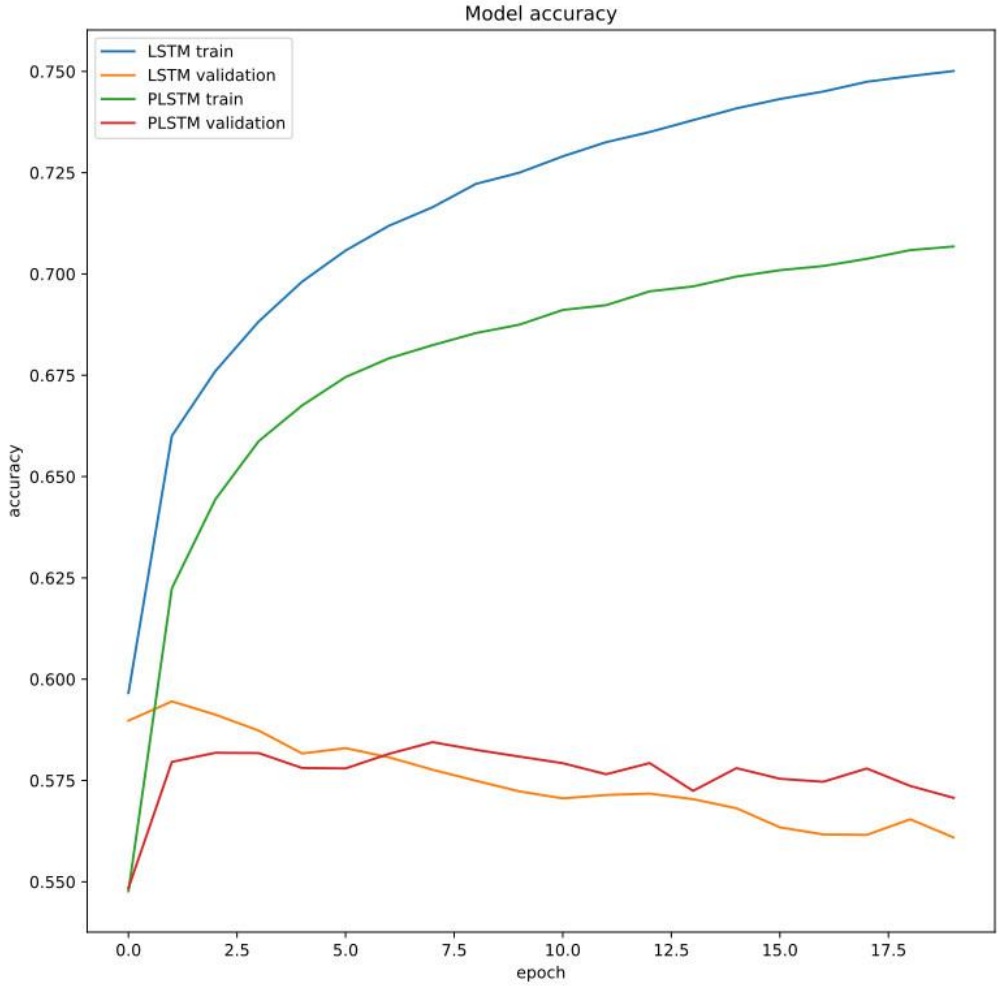


Model Modification

Drop-out rate 0.5

Increased max sequence length

Less neurons in hidden layer



Conclusions

The less complex LSTM model provides better results.

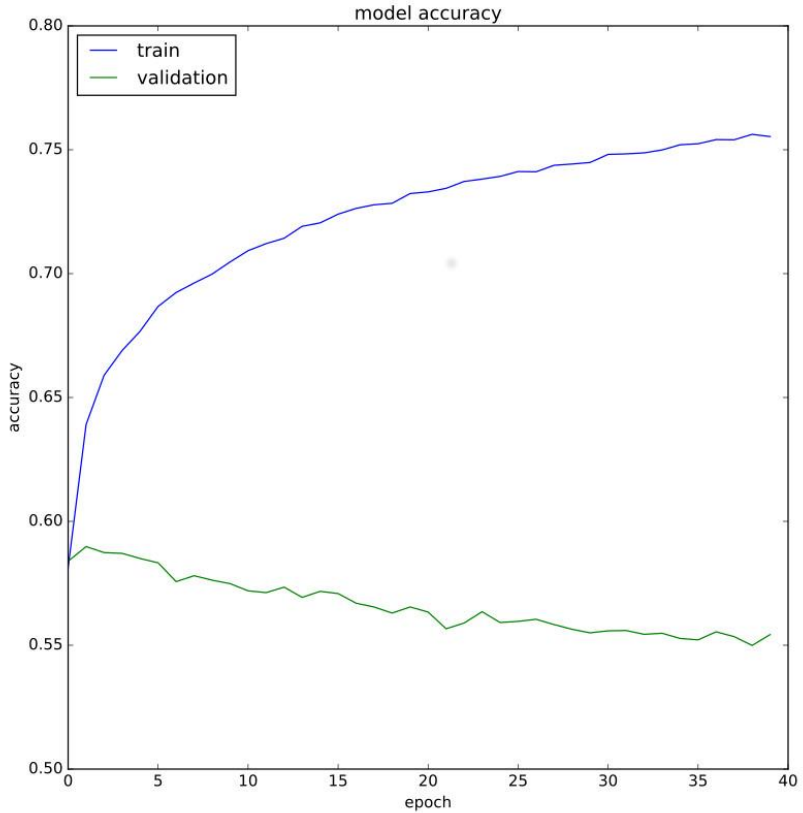
To avoid overfitting the data set, early stopping is the more successful strategy:
Optimal results are obtained after 5-10 epochs!

Increasing the drop-out ratio:
- increases temporal variance of model performance (especially for PLSTM),
- stabilizes model performance but does not increase it.

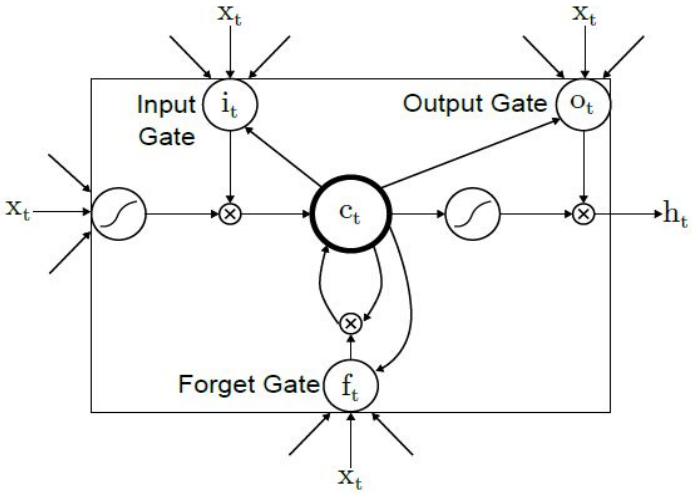
Future work:
Find models with an explicit regularization parameter.
Decrease the learning rate.

Appendix

LSTM with one additional laver and 20% drop-out rate



Model details: LSTM Cell



Source: Neil, D., Pfeiffer, M., & Liu, S. (2016). Phased LSTM : Accelerating Recurrent Network Training for Long or Event-based Sequences arXiv : 1610 . 09513v1 [cs . LG] 29 Oct 2016, (Nips).

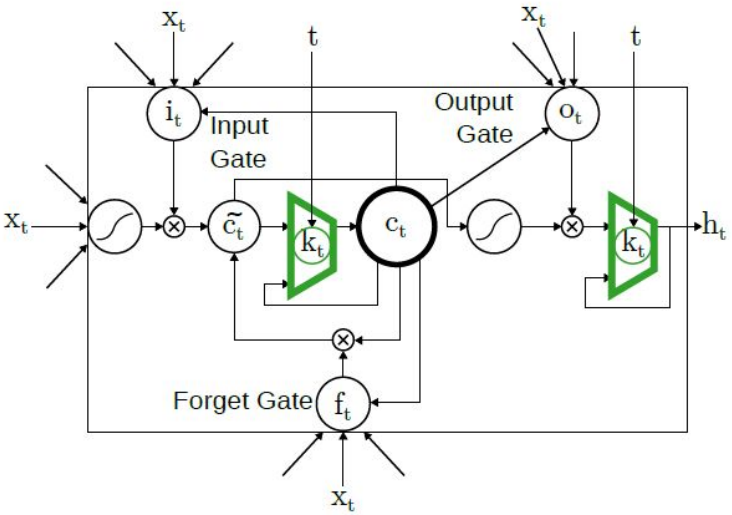
LSTM blocks contain three or four "gates". These gates implement logistic functions to compute a “weight” [0,1]. This allows to control the flow of information into or out of their memory:

- An *"input gate"* controls the extent to which a new value flows into the memory.
- A *"forget gate"* controls the extent to which a value remains in memory.
- An *"output gate"* controls the extent to which the value in memory is used to compute the output activation of the block.

The key points of an LSTM RNN:

- **No activation function** within its recurrent components. (This avoids gradient vanishing overtime due to backpropagation).

Model details: Phased LSTM Cell



Source: Neil, D., Pfeiffer, M., & Liu, S. (2016). Phased LSTM : Accelerating Recurrent Network Training for Long or Event-based Sequences arXiv : 1610 . 09513v1 [cs . LG] 29 Oct 2016, (Nips).

Current RNN models are not well suited to process **irregularly sampled data** triggered by events generated in continuous time by sensors or other neurons.

- *Phased LSTM* was introduced building on top of existing LSTM cell an additional *"time gate"* which allows the LSTM cell to be actuated at timestamp *k_t* (asynchronous control).
- In the Phased LSTM formulation, the cell value *c_t* and the hidden output *h_t* can only be updated during an "open" phase; otherwise, the previous values are maintained.

The key point of Phased LSTM:

- It introduces time dependency actuation to allow **asynchronous inputs**.

Bibliography

- Neil, D., Pfeiffer,, M., & S., Liu. (2016). Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences. *Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences*. Retrieved January 30, 2017.
- Socher, R., Perelygin, A., Wu, J. Y., Chuang,, J., Manning, C. D., NG, A. Y., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. Retrieved January 30, 2017.
- Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05*. doi:10.3115/1219840.1219855
- Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting.*Journal of Machine Learning Research*, *15*(1), 1929-1958.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.