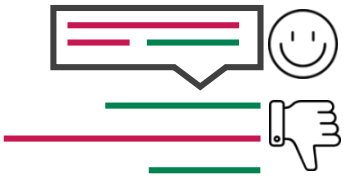


PRIM Project 2017

Sentiment Analysis with multiplicative Long Short Term Memory Network

Wei Chen

Supervised by Pierre-Yves Casanova, Chloé Clavel

3rd February, 2018

CONTENTS

1 Introduction

1.1 Problem statement

Sentiment analysis, also known as opinion mining, is the study which analyzes people's opinions, sentiments, subjectivity, evaluations, attitudes, appraisal, affects, views, emotions expressed in text [1]. One of the common applications of sentiment analysis is the emotion detection from product or service reviews like in Amazon or Allociné. A Chinese online shopping website *taobao.com* has already implemented automatic emotion extraction from customers' reviews. Based on its extracted information, retailers can get prompt feedback of the satisfactory from the customer for a certain purchase.

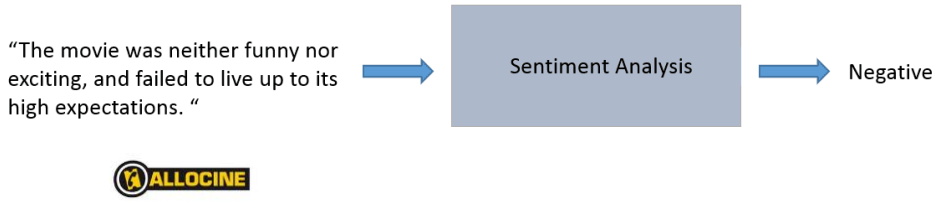


Figure 1. A typical application of sentiment analysis system, evaluating movie reviews on *allocine.fr*

Besides this, there are a wide range of tasks that sentiment analysis can deal with, however in this project we only focus on two most basic and common ones.

- **Polarity detection**: Polarity detection assumes that the sentiment in all text can be classified as two poles, so it simply tries to find whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral.
- **Emotion analysis**: Emotion analysis goes beyond polar sentiments. Instead of focusing only on positive/negative, it has more sophisticated aim of detecting different emotional states, such as "angry", "sad", "happy" etc. expressed in the text.

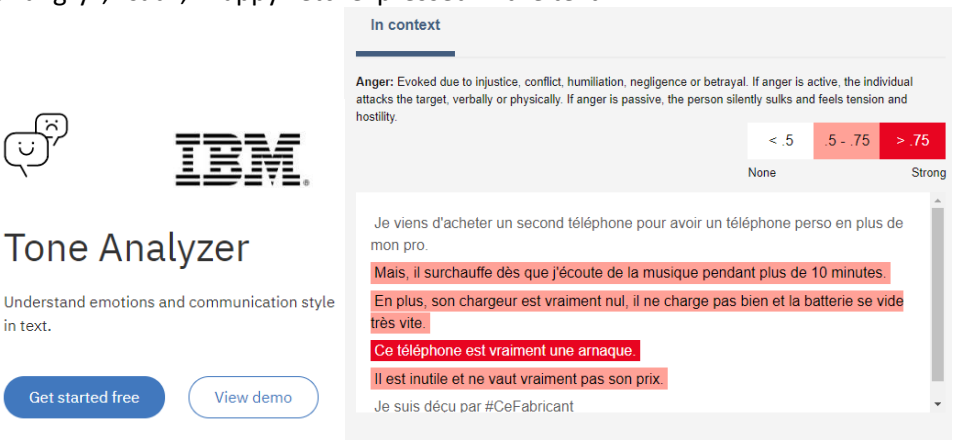


Figure 2. Tone Analyzer developed by IBM's DeepQA project, emotion analysis toolkit based on n-gram features and SVM classifier. It allows to detect 4 emotions including joy, fear, sadness, anger, as well as 3 communication style including analytical, confident and tentative found in text.

A naïve approach for sentiment analysis is to detect the opinion words in the text, then use some fixed rules to determine the expressed sentiment. However there are a few problems that make it specifically hard:

- Negations

  A classic argument for why using a bag of words model doesn't work properly for sentiment analysis. *"I like the product"* and *"I do not like the product"* should be opposites. A classic machine learning approach would probably score these sentences identically.

- Metaphors, Irony, Jokes

  Computers always have trouble understanding figurative language. *"The best I can say about this product is that it was definitely interesting…"* Here, the word *"interesting"* plays a different role than the classic, positive meaning.

- Multiple sentiments in the same text

  A complex text can be segmented into different sections, while some sections can be positive, others negative. For example, "*The phone's design is the best I've seen so far, but the battery can definitely use some improvements*", here we can see the presence of two sentiments. It will be very complicated to classify the review as positive or negative.

1.2 Sentiment analysis on different levels

According to the different scale of text that we study, sentiment analysis has been categorized mainly into three levels:

- **Document level**: The task at this level is to classify whether a whole opinion document expresses a positive or negative sentiment [4,5]. For example, given a product review, the system determines whether the review expresses an overall positive or negative opinion about the product. This level of analysis assumes that each document expresses opinions on a single entity (e.g., a single product). Thus, it is not applicable to documents which evaluate or compare multiple entities.
- **Sentence level**: The task at this level goes to the sentences and determines whether each sentence expressed a positive, negative, or neutral opinion.
- **Aspect level**: Both the document level and the sentence level analyses do not discover what exactly people liked and did not like. Based on the idea that an opinion consists of a sentiment and a target of opinion, the aspect level sentiment analysis detects the sentiment towards each aspect or feature of an entity. For example, although the sentence "*although the service is not that great, I still love this restaurant*" clearly has a positive tone, we cannot say that this sentence is entirely positive. In fact, the sentence is positive about the restaurant (emphasized), but negative about its service (not emphasized).

1.3 Application examples

Opinions are essential to almost all human activities because they are key influencers of our behaviors. Especially with the explosive growth of social media (online reviews, forum discussions, blogs, micro-blogs, Twitter, comments, ratings) on the Web, individuals and organizations are increasingly using the content in these media for decision making. We try to enumerate some typical applications of sentiment analysis. With the expansion of social media in people's life, there must be more new merging areas where sentiment analysis can be applied, so this list is absolutely not exclusive.

- **Review-related websites**

  For review-related sites like Amazon or IMDb, finding and monitoring opinions and distilling the information contained in reviews remains a formidable task because of the proliferation of diverse sites. The average human reader will have difficulty in identifying, extracting and summarizing the opinions in them. Automated sentiment analysis systems are thus needed.

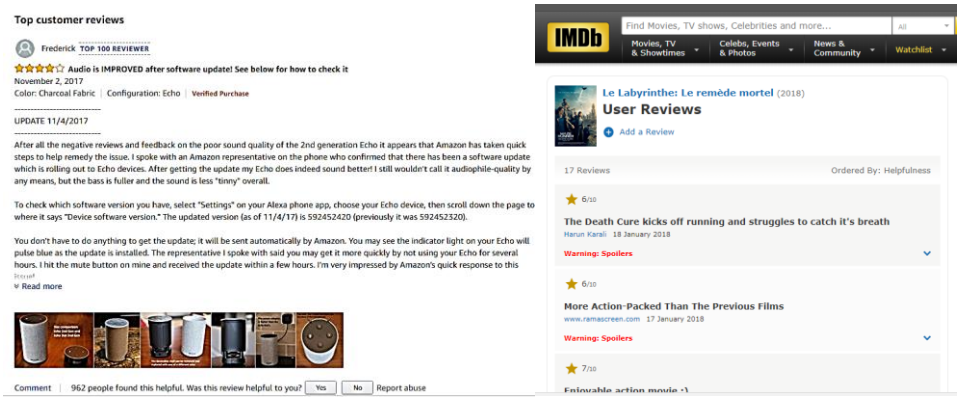


Figure 3. Samples of Amazon review and IMDb reviews

- **Ad placement**

  In online systems that display ads as sidebars or banners, it is helpful to detect webpages that contain sensitive content inappropriate for ads placement [7]. For more sophisticated systems, it could be useful to bring up product ads when relevant positive sentiments are detected, while more importantly, block the ads when relevant negative statements are discovered.

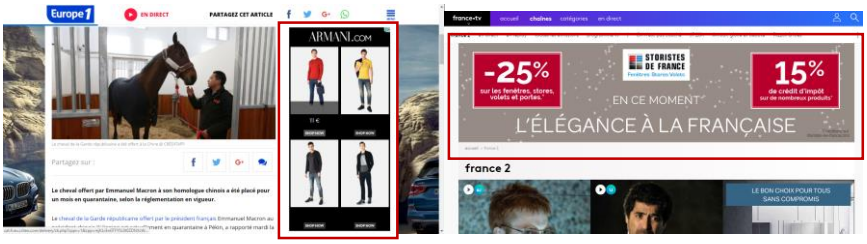


Figure 3. Examples of webpage with content ad

- **Reputation management and public relations**

  Companies, organizations and governments perhaps hope that by tracking public viewpoints, they could monitor sources for increases in hostile or negative communications in social media or news press. On the other hand many organizations intend to use their internal data, e.g., customer feedback collected from emails, call centers or surveys to monitor potential risks and prevent reputation loss.

1.4 Tasks of this project

Because of all the possible applications, there are a good number of companies, large and small, that take opinion mining and sentiment analysis as part of their mission. This project proposed by BNP Paribas CIB has two missions:

- **Negative consensus monitoring**

  BNP intends to monitor the consensus from opinions and reviews in news and social media. These opinions and reviews may be towards BNP or its clients. Some of them are positive signals to the bank, while others can be potential risks. Since manually classifying of these comments or opinions can be a huge work, no mention monitoring the consensus in press in real time, the company has to turn to a more intelligent and efficient way.

- **Analysis of traders' emotion**

  As a global investment bank, BNP handles thousands of trades in including stock, commodity, etc. Many banks have been heavily fined by regulators in the past years due to market abuse by some of their employers, so that the bank has to establish a mature system to supervise traders' behavior. BNP has already a large number of chat history, email, phone call transcripts

between traders on shelf, and what they need is to take advantage of these data and analyze traders' emotions while purchasing and selling in order to identify potential market abuse situations.

2 Review of approaches

Although linguistics and natural language processing (NLP) have a long history, little research had been done on this topic before the year 2000. The research on sentiments and opinions appeared in 2001[2-6]. Since then, the field has become a very active research area.

In the literature on this domain, sentiment analysis techniques can be roughly divided into machine learning approach, lexicon based approach and hybrid approach. The machine learning approach (ML) applies ML algorithms and uses linguistic features, while the lexicon-based approach relies on a sentiment lexicon, a collection of known and precompiled sentiment terms. The hybrid Approach combines both approaches and is very common with sentiment lexicons, playing a key role in the majority of methods. The various approaches of sentiment analysis and their categorization are illustrated in figure 4.

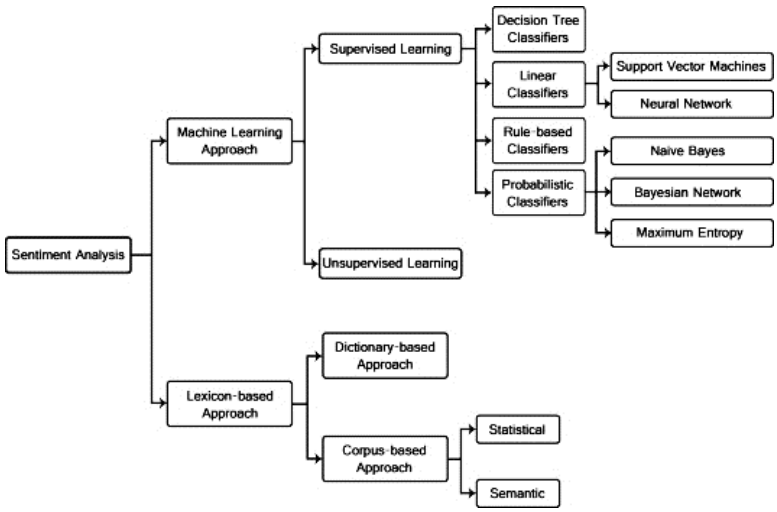


Figure 4. Sentiment classification techniques [23]

2.1 Lexicon/symbolic-based approach

Even before machine learning, engineers interested in classifying sentiment would employ heuristics like keyword search to get the job done. The lexicon-based approach depends on finding the opinion lexicon which is used to analyze the text. It can be further categorized into following two kinds:

- **The dictionary-based approach** which depends on finding opinion seed words, and then searches the dictionary of their synonyms and antonyms.
  Qiu and He [24] used this approach to identify sentiment sentences in contextual advertising (i.e. associating ads with a relevant web page). By using syntactic parsing and a sentiment dictionary, they proposed a rule based approach to extract topic words of opinion sentences associated with negative sentiment. The sentiment dictionary they utilized is called General Inquirer consists of 1892 negative words and 1563 positive Their work was applied to online forums automotvieforums.com and demonstrated effectiveness on advertising keyword extraction and ad selection.
  There are more than one sentiment dictionary being used for sentiment analysis. For instance the well-known Python package nltk provides SentiWordNet [38] a lexical resource for opinion mining on the basis of WordNet. It assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. Also we have WordNet-Affect [29], another extension of

WordNet, including a subset of synsets suitable to represent affective concepts correlated with affective words. These developed dictionaries have been proved effective for certain tasks of sentiment analysis and certain datasets. However it's obvious that the dictionary based approach has a major disadvantage of being unable to mine the opinion with domain and context specific orientations. A sentence like, "*I hope you were happy*," could easily be misinterpreted as having a positive connotation simply because it possesses the word happy.

- **The corpus based approach** begins with a seed list of opinion words, and then finds other opinion words in a large corpus to help finding additional opinion words with oriented context. One example of this method was presented by Hatzivassiloglou and McKeown [25]. They started with a list of adjective with seed opinion, and used them along with a set of linguistic constraints to identify additional adjective opinion words and their orientations. The constraints are connectives like *AND, OR, BUT, EITHER-OR*…In order to determine if two conjoined adjectives are of the same or different orientations, learning is applied to a large corpus. Then, the links between adjectives form a graph and clustering is performed on the graph to produce two sets of words: positive and negative.
  Nevertheless, using the corpus-based approach alone is not as effective as the dictionary-based approach because it is hard to prepare a huge corpus to cover all English words, but this approach has a major advantage that considers domain information and context specific opinion words by using domain corpus.

2.2 Supervised learning approach

The supervised learning methods depend on the existence of labeled training documents. There are many kinds of supervised classifiers applied in the literature on sentiments analysis. Most frequently used classifiers in sentiment analysis include naive Bayes classifier, Bayesian network, SVM, neural network, decision tree classifier. Table 1 is an incomplete list of researches using supervised learning approach for sentiment analysis.

Table 1. Examples of supervied learning approaches applied to sentiment analysis

	domain	*classes*	*performance*
Blair-Goldensohn et al. (2008)	restaurant & hotel reviews	Polarity detection	precision:68.0%/77.2% recall: 90.7% / 86.3%
Yu et al. (2011)	product reviews	Polarity detection	F1: 71.7%-85.1%
Choi & Cardie (2008)	MPQA corpus [20]	Polarity detection	accuracy: 90.70%
Lu et al. (2011)	restaurant & hotel reviews	5-star rating	LAE: 0.560 - 0.790
Titov & McDonald (2008)	product, hotel, restaurant reviews	Polarity detection	Ranking Loss: 0.669

2.3 Unsupervised learning approach

Large number of labeled training documents are used for supervised learning, as stated before. In text classification, it is sometimes difficult to create these labeled training documents, but it is easy to collect the unlabeled documents. The unsupervised learning methods overcome these difficulties.

For example in the work of Ko and Seo [26], they proposed a method that tokenizes the documents into sentences, and categorized each sentence using keyword lists of each category and sentence similarity measure.

The unsupervised approach was used by Xianghua and Guo [27] to automatically discover the aspects discussed in Chinese social reviews and also the sentiments expressed in different aspects. They used LDA model to discover multi-aspect global topics of social reviews, then extracted the local topic and associated sentiment based on a sliding window context over the review text. They showed that their approach obtained good topic partitioning results and helped to improve sentiment analysis accuracy. It helped too to discover multi-aspect finegrained topics and associated sentiment.

3 From LSTM to mLSTM

Labeled data are the fuel for today's machine learning. As we mentioned in section 2.3, collecting data is easy, but scalably labeling that data may be extremely hard. So Machine learning researchers have long dreamed of developing unsupervised learning algorithms to learn a good representation of a dataset, which can then be used to solve tasks using only a few labeled examples[8]. This is also the motivation of this project to find unsupervised approach for sentiment analysis problem.

In this section we will turn to an unsupervised model called mLSTM. This model has been found very useful in language modeling where it is used only to predict the next character in text sequence. The model's capability of efficient sentiment representation has been found coincidentally and soon used for sentiment analysis [15]. To clearly introduce the architecture of mLSTM and its advantage, we will start from regular LSTM, then see how the mLSTM modifies the architecture on the basis of LSTM. Finally we will explain how the model can be used in language modeling and text representation that is effective for sentiment analysis.

3.1 LSTM

Long Short Term Memory networks (LSTM) are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter and Schmidhuber in 1997 [9], and were refined and popularized by many researchers.

In NLP the model's capability of connecting previous information to the present task is very important. Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist. However when the gap between the relevant information and the point where it is needed becomes very large, RNN may lose its appeal. In this case we need more context, and LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn! Andrej Karpathy et al showed in their paper that LSTMs can keep track of text attributes like quotes, line lengths and brackets in interpretable cells [10].

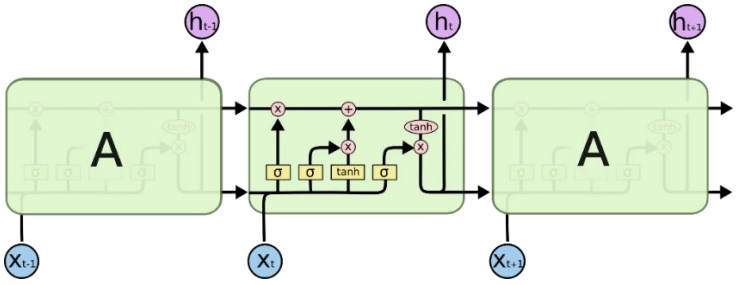


Figure 5. The repeating module in a LSTM

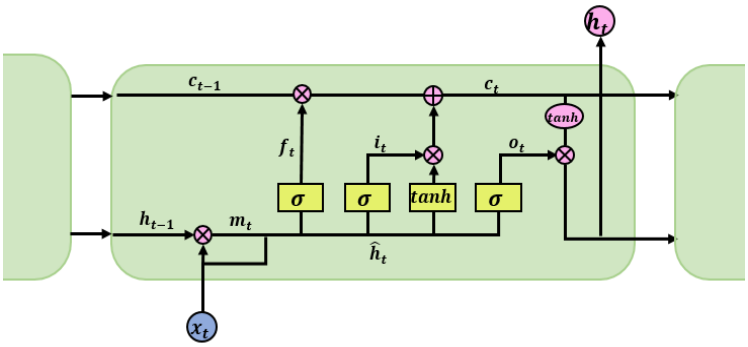
Like normal RNN, LSTMs also have the chain structure where four layers interact in a special way help the network memorize long term information. In figure 5 we will illustrate this structure in the hidden layers.

3.2 mLSTM

Generative RNNs are known to have problems with recovering from mistakes [11]. Each time the recursive function of the RNN is applied and the hidden state is updated, the RNN must decide which information from the previous hidden state to store. If the RNN's hidden representation remembers the wrong information and reaches a bad numerical state for predicting future sequence elements, for instance as a result of an unexpected input, it may take many time-steps to recover.

To resolve this fatal flaw of RNN, the input-dependent transition functions of RNN was proposed to make the network recover from unexpected input faster. This is the key idea of multiplicative RNN (mRNN) [12], which is designed specifically to allow flexible input-dependent transitions.

Multiplicative LSTM (mLSTM) is the hybrid architecture that combines the factorized hidden-to-hidden transition of mRNNs with the gating framework from LSTMs [13]. As a variant of LSTM, the mLSTM just has an extra intermediate state m_t connecting to each gating units in the LSTM. The resulted architecture is illustrated in the following figure:



$$m_t = (W_{mx}x_t) \odot (W_{mh}h_{t-1})$$
$$\hat{h}_t = W_{hx}x_t + W_{hm}m_t$$
$$i_t = \sigma(W_{ix}x_t + W_{im}m_t)$$
$$o_t = \sigma(W_{ox}x_t + W_{om}m_t)$$
$$f_t = \sigma(W_{fx}x_t + W_{fm}m_t)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(\hat{h}_t)$$
$$h_t = tanh(c_t) \odot o_t$$

Figure 6. The modified architecture of mLSTM by adding an intermediate state m_t

Due to the higher speed of recovery from unexpected input, mLSTM has been proved to converge faster with lower loss than regular LSTM on character level language models, which is verified by Krause et al [13] by training both models for sequence prediction.

3.3 LSTM/mLSTM in language modelling

LSTM network is commonly used in text prediction because of its good nature of memorizing long term dependency. Text prediction aims to predicting the next word or character in a sequence of text. More formally, given a training sequence $(x_1, \ldots, x_t)$, the LSTM uses the sequence of its output hidden states $(h_1, \ldots, h_t)$ to predict the next coming character with a softmax layer or a linear function [14]. In our work we use the following linear function where y_t denotes the t-th character in a sequence.

$$y_t = W_{yh}h_t$$

Then the cost function in the model can be defined as the cross entropy loss

$$loss(y_t, x_{t+1}) = -y_t(x_{t+1}) + log\left(\sum_j exp(y_t(j))\right)$$

The language modeling objective is to minimize the training loss of the training sequence. For this optimization problem, the backpropagation algorithm is widely used for model training. So the full architecture of language modeling using mLSTM can be illustrated as follows.

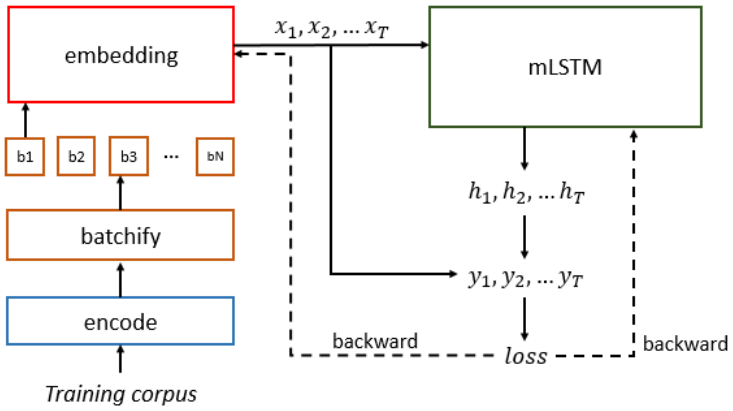


Figure 7. Training mLSTM for text prediction

First the training corpus is encoded into byte representation where each character is encoded by an integer. Then according to batch size the whole encoded corpus will be partitioned into a certain number on batches. These text batches are sequentially fed to the embedding layer where every byte in every batch will be mapped into a vector of fixed length. These vectors $(x_1, \ldots, x_t)$ are exactly the sequential inputs of our mLSTM module, and through the forward propagation it will output the hidden states h_t and finally the predicted character y_t at each time step. Finally based on the cross entropy $loss(y_t, x_{t+1})$, by using the backpropagation we are able to update the parameters of both mLSTM model and embedding model.

3.4 mLSTM for sentiment representation

With the trained mLSTM model, not only can we obtain an efficient generative model for text prediction, but also this model can be applied in sentiment analysis. Radford et al [15] find this generative model learns an excellent representation of sentiment, despite being trained only to predict the next character. They even find there actually exists a single "sentiment neuron" in the mLSTM units that is highly predictive for the sentiment value. For polarity detection this "sentiment neuron" allows us to classify the text with unsupervised method, for example simply by thresholding the sentiment neuron value to determine its polarity. This is how the mLSTM model outperforms other supervised model, because of its larger universality especially when we don't have much (or even any) labeled data.

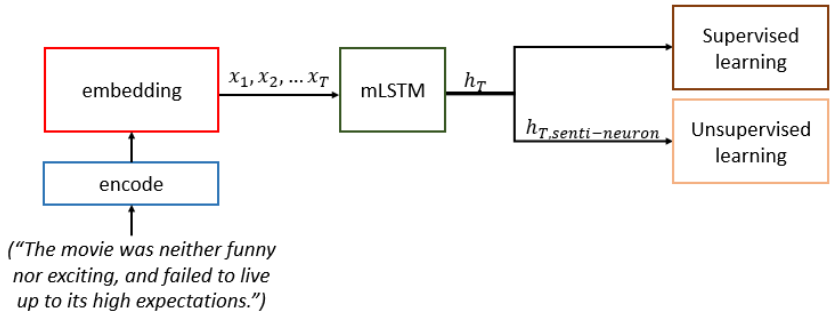


Figure 8. mLSTM for sentiment representation

So in our work we will take advantage of mLSTM model to learn an unsupervised text representation which can accurately reveal the expressed sentiment. Then on the top of this representation we will respectively test supervised and unsupervised methods for sentiment classification and their capacity celling.

4 Experiment

In this section we will firstly try several baseline methods other than mLSTM on several widely used sentiment analysis datasets. Then we will turn to the more sophisticated mLSTM model we introduced in section 3.

4.1 Baseline test

4.1.1 SentiWordNet

WordNet is a large lexical database where nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. And the synsets are interlinked by means of conceptual-semantic and lexical relations. SentiWordNet [38] is just an extension of WordNet. It assigns to each synset (117,659 synsets in total) of WordNet three sentiment scores: positivity, negativity, objectivity which thus are very useful for opinion mining.

In the first baseline that we implement in our experiment, we use the SentiWordNet module in nltk toolkit to detect sentiment polarity simply by searching each word in the synset and determining the polarity according to the sum sentiment scores of all words. This is a lexicon-based method without any supervision of labeled text.

Figure 9 will show how we preprocess the text and feed it to SentiWordNet. For instance, a sentence "*The movie was neither funny nor exiting*." will be first tokenized into a list of words, then we mark every words in the sentence a part of speech tag, besides mark every word after the negation word a "*NEG*" tag. The POS tag will be important for the lemmatization, and the "*NEG*" tag will help determine the sentiment polarity of a word in the context. Then we remove the words other than nouns, verbs, adjectives and adverbs, and find their lemmatized form. In our example this will result in *('movie', 'NN'), ('be', 'VBD'), ('funny','NEG', 'JJ'), ('exciting','NEG', 'JJ')*. We search these 4 synset along with their tags in the SentiWordNet dictionary to find the positive and negative score of each. The final polarity will be determined as follows

$$polarity = sign(\sum_i \alpha_i \cdot (posscore_i - negscore_i))$$

where the $\alpha_i = \begin{cases} -1, if\ NEGtag = 'NEG' \\ 1, if\ NEGtag = None \end{cases}$, and i denotes the i-th lemmatized word.

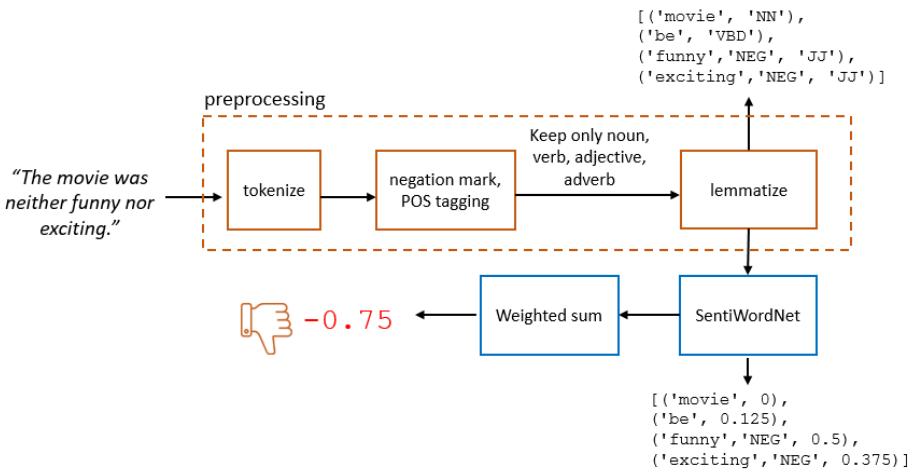


Figure 9. SentiWordNet used as a baseline for sentiment analysis

We test this approach on 4 middle scale of datasets in which

- IMDb dataset [32] consists of 50,000 IMDB movie reviews, specially selected for sentiment analysis. The sentiment of each reviews is binary and balance, meaning the IMDB rating < 5 results in a sentiment score of 0, and rating >=7 have a sentiment score of 1, and the proportions are both 50%.
- Stanford Sentiment Treebank [19] are review sentences extracted from movie review websites *rottentomatoes.com* with binary labels on sentiment, The label is based on the average rating of movies and relabeled via Amazon Mechanical;
- MPQA2.0 contains 692 news articles from 187 different foreign and U.S. news sources [20]. Among them 8016 phrases are annotated as subjective with a. contextual polarity value. We select 7977 of them in the experiment and treat the binary form of contextual polarity as sentiment labels of the sentence.
- Twitter2016 binary is the tweets collection selected from tweets posted in 2016, provided by SemEval-2017 Task 4 [30]. It contains 28,613 tweets, and each of them is annotated with a sentiment label positive/negative/neutral. Here we use its binary form by eliminating all the neutral samples, and preprocess them by deleting all @, # tags and url links.

The test results are presented in table 2.

4.1.2 n-gram

As discussed in section 2.1, lexicon-based method is high biased because of the complex context possibly. For instance with the model based on SentiWordNet, a sentence as simple as "*The movie(0) did(0)n't live(0) up(0) to high(+0.125) expectation(0).*" will be classified as positive, because SentiWordNet failed to regard "live up to" as a whole phrase.

In NLP this problem can be successfully resolved by n-gram. An n-gram is a contiguous sequence of n items from a given sample of text. One main benefit of n-gram models is its ability to store more context with larger n. The n-gram is used as an important feature for text data in text classification. For sentiment analysis A. Pak et al [39] used n-gram feature in their work to classify sentiments in tweets. Here we propose anther baseline by using unigram and bigram as the feature of SVM classifier, and do the polarity detection with supervised training. Again we test on the 4 datasets we mentioned in 4.1.1, and the performance is shown in table 2.

Table 2. Baselines perforamance, SentiWordNet and n-gram

dataset	*scale*	*SentiWordNet*	*unigram+linear SVM*	*unigram+bigram +linear SVM*
IMDb[1]	50,000 documents	66.4%	86.7%	**88.6%**
Stanford Sentiment Treebank[2]	9,613 sentences	59.9%	78.1%	**79.8%**
Twiiter2016 binary[3]	15,581 sentences	69.2%	73.7%	**73.8%**
MPQA2.0[4]	7,977 sentences	60.0%	81.7%	**82.6%**

[1] http://ai.stanford.edu/~amaas/data/sentiment/

[2] https://nlp.stanford.edu/sentiment/treebank.html

[3] http://alt.qcri.org/semeval2017/task4/index.php?id=data-and-tools

[4] http://mpqa.cs.pitt.edu/corpora/mpqa_corpus/mpqa_corpus_2_0/

We see from the results, SentiWordNet turns out to be a very naïve method with relatively low precision over all the four datasets. However the SVM classifier training on n-grams seems to be much more effective, especially for IMDb and MPQA2.0 dataset.

4.2 mLSTM model for sentiment analysis

As stated in section 3, our target of this project is to seek unsupervised methods which can be apply for sentiment analysis by using only none or a few labeled examples. Thus, although the strongly supervised method like SVM on n-grams work very well in some datasets, it is hard to use when we want to extend the method to other domain where there are few labelled data.

To this end, we continue to turn to the mLSTM we introduce in section 3. In the following part we will firstly introduce the general settings to train an mLSTM model on the Bloomberg datasets. Then we will apply the pretrained model for sentiment analysis task and evaluate it on various widely used datasets in both supervised and unsupervised way. Moreover we will train a new model based on different type of training corpus, which will enable the model to deal with more complicated and more various types of texts. Finally we will propose a generative model which can be applied to generate text with customized sentiment polarity.

4.2.1 Training on Bloomberg news corpus

Following the schema illustrated in figure 7, and using backpropagation through time (BPTT) algorithm [16], we train our mLSTM model on the Bloomberg corpus[5]. This corpus contains 450,341 news articles of size 1.27GB by Bloomberg L.P. from October 2006 to November 2013. Bloomberg is financial news and media company based in New York. Since we mentioned in section 1.4 that one of the tasks of this project is to monitor the negative consensus in media. We intend to obtain a model which is more familiar with the news context particularly the financial news. That's why we choose Bloomberg corpus as our training set.

Before fed into the training model, the training corpus has been preprocessed by the following three steps:

- All newline characters within documents are replaced with spaces.
- Any leading whitespace is removed and replaced with a newline+space to simulate a start token. Any trailing whitespace is removed and replaced with a space to simulate an end token.
- The text is encoded as a UTF8 byte sequence. So different from common sentiment analysis we introduced in section 1.3, this one will be on character level.

The model training is based on Pytorch and cuda 8.0 on a single 11GB-memory Tesla K80 GPU. To make the new model compatible with the one in [15], we try to keep the training parameter as the same, with sequence length=256, batch size=32, embedding size=64, number of layers=1, number of RNN units=4096. And we split the whole dataset into 1002 equal shards and set aside 1 shard for validation and 1 shard for test. States were initialized to zero at the beginning of each shard and persisted across updates to simulate full-backpropagation and allow for the forward propagation of information outside of a given subsequence. The model results in 86 million weights to update, and finally it took approximately 12 days to train 1 epoch in total with learning rate 0.000125.

During the entire training process, we tracked the cross entropy loss across 142,237 batches. Also we compute the average loss during the training simply by smoothing the cross entropy loss curve:

$$avg\_loss(n) = 0.99avg\_loss(n-1) + 0.01loss(n)$$

[5] https://github.com/philipperemy/financial-news-dataset

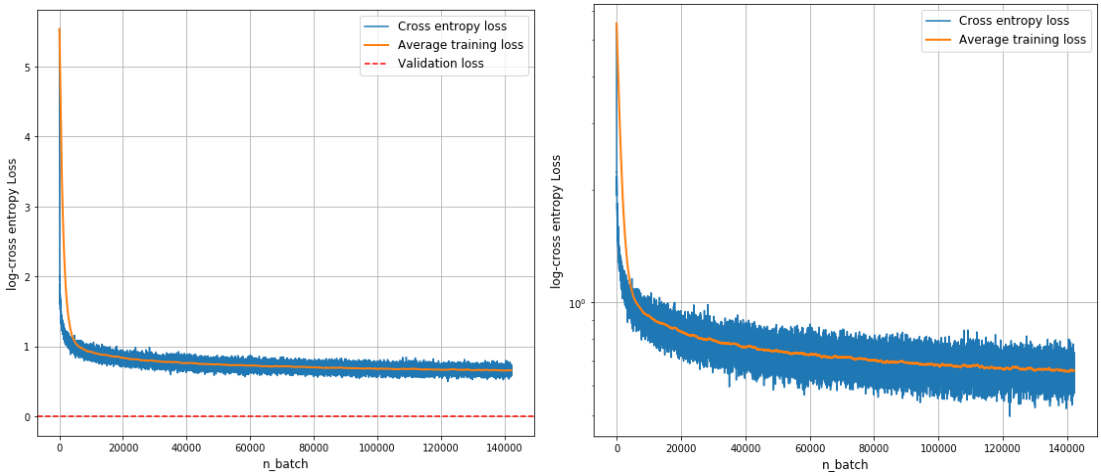


Figure 10. Cross entropy loss and average loss across the entire 142,237 batches respectively in linear scale and log scale. The model almost converged after being trained with first 20,000 batches, but still optimizes itself by slight step afterwards. The training ends up with validation loss 0.00016 after 1 epoch.

4.2.2 Model test with supervised learning

With the trained mLSTM model we are now able to learn the character level representation based on this model. But before inputting the test data to the model, we need the same preprocessing for test samples as training corpus. By following the schema in figure 8, model states are initialized to zeros. Then for each byte in a sample, the model updates its hidden state and predicts a probability distribution over the next possible byte. The hidden state of the model serves as an online summary of the sequence which encodes all information the model has learned to preserve. The preserved information will facilitate to predict the future bytes of the sequence.

The final cell states of the mLSTM are used as a feature representation. Then a Tanh layer is applied to bound values between -1 and 1. This representation will encode each text sample with a vector whose length is the hidden state size of the mLSTM module (4096 in our training set).

Then on top of our model's representation we train a logistic regression classifier with l_1 penaltiy on labelled datasets. The objective function of a l_1 regularized logistic classifier is

$$argmin_w \frac{1}{n}\sum_{i=1}^{n} log\left(1 + exp\left(-p_i\left(h_i^T w + w_0\right)\right)\right) + C\|w\|_1$$

where h_i is the final hidden state of i-th training sample, p_i is the polarity label of i-th training sample, w it the weights to learn.

We split the whole dataset into 3 shards respectively for training (70%), validation(20%) and test(10%), and use cross validation to optimize the regularization parameter C within logistic regression. Table 2 shows the test results of the Bloomberg model on several labelled standard datasets for polarity detection. 4 of them were mentioned in the baseline test, the others are:

- the Rotten Tomatoes [18] and Stanford Sentiment Treebank [19] are sentences both extracted from movie review websites *rottentomatoes.com* with binary labels on sentiment, however the latter one was relabeled via Amazon Mechanical;
- Amazon datasets [28] are sentences extracted from Amazon reviews of five electronics products: 2 digital cameras, 1 DVD player, 1 mp3 player, and 1 cellular phone;
- Yelp reviews dataset is obtained from the Yelp Dataset Challenge in 2015. This dataset contains 1,569,264 samples of customers' reviews mostly on restaurants. The polarity dataset has 280,000 training samples and 19,000 test samples in each polarity.

- OpinMind ***[29]*** is the deduplicated form of comments on the online forum opinmind.com.

In addition to the model trained on Bloomberg corpus, we also include the test results of the pretrained model provided in [15]. This model was trained on Amazon product review corpus introduced in [17]. In de-duplicated form, this dataset contains over 82 million product reviews from May 1996 to July 2014 amounting to over 38 billion training bytes (38GB).

Table 3. Performance of pretrained model on standard polarity detection datasets

Dataset	*Genre*	*Scale (# of sentences)*	*Distribution pos/neg*	*Accuracy with Bloomberg model*	*Accuracy with Amazon model*	*benchmark*
Rotten Tomatoes[18]	Movie review	10,662	0.5/0.5		**86.9%** [15]	83.1%[34]
Stanford Sentiment Treebank[19]	Movie review	9,613	0.52/0.48		**91.8**%	90.2%[35]
IMDb[32]	Movie review	50,000 (documents)	0.5/0.5		92.3%	**94.1% [36]**
Amazon[28]	Product review	3,788	0.64/0.36		**91.4%** [15]	86.3%[34]
Yelp[6]	Restaurant review	598,000 (documents)	0.5/0.5		95.2% [15]	**95.6%** [33]
MPQA2.0 [20]	News article	7,977	0.32/0.68		81.1%	**93.3%** [34]
Twitter2016[30]	Social media	28,631	0.38/0.16/ 0.46(neutral)		54.7%	**65.1%** [37]
Twiiter2016 binary	Social media	15,581	0.71/0.29		84.5%	----
OpinMind[7]	Online forum	34,462	0.55/0.45		92.6%	**93.8% [29]**

The results suggest that the Amazon model has learned a rich representation of text from product reviews. That is why the classification results on amazon product review datasets outperform the existing benchmark 86.3% by ADASENT [21].

However for other type of text like tweets or news articles, due to the limit of vocabulary, sentence structure and subjects, the Amazon model turns out be not as good as on reviews although it is trained on the corpus as large as 38 GB.

4.2.3 Sentiment neuron

In last subsection, we apply a l_1 regularized logistic regression classifier on top of the sentiment representation. The l_1 regularization is known to reduce sample complexity when there are many

[6] https://www.yelp.com/dataset/challenge
[7] https://www.kaggle.com/c/si650winter11/data

irrelevant features. This is likely to be the case for our model since it is trained as a text predicting model instead of a supervised sentiment extractor. If we observe the resulted weights *w* of the trained logistic classifier, in figure 11 we can easily notice one certain weight stands out with very high value compared to all others, especially for Twitter2016 binary dataset and SST dataset. This result means there is a single unit within the mLSTM that almost directly correspond to the sentiment.

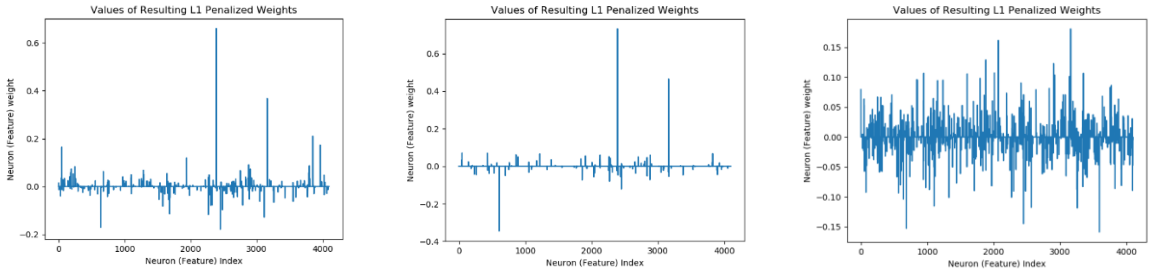


Figure 11. Weights *w* of the logistic classifier trained respectively on Twitter2016 binary, SST and MPQA2.0. The 2388th feature turns out to be the sentiment neuron.

Figure 12 shows the histogram of sentiment value of all positive and negative samples in SST and opinmind test set. We see the histogram follow a bimodal distribution with clear separation between the positive and the negative. Even more we can take advantage of this sentiment neuron to visualize the sentiment in the test at byte level. In figure 13, we selected two pieces of review from IMDb and Amazon, and we colored the text according to the sentiment neuron value resulted by each byte, from green to red signifies from positive to negative.

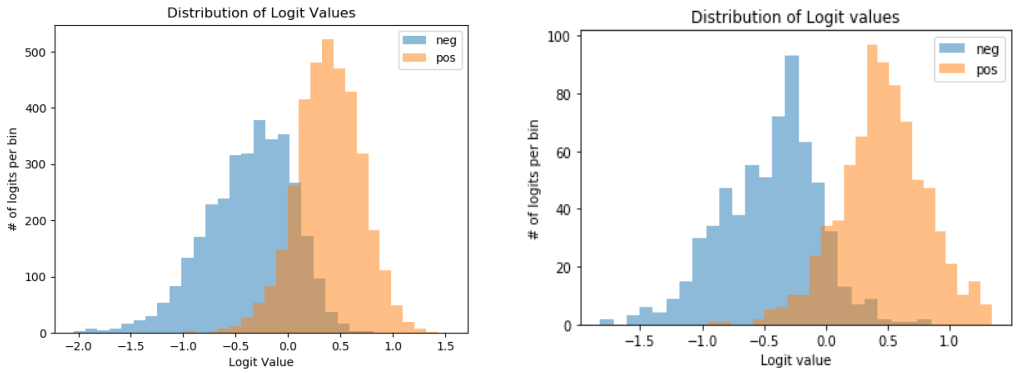


Figure 12. Histogram of sentiment value, respectively on of SST and opinmind test set

25 August 2003 League of Extraordinary Gentlemen: Sean Connery is one of t
he all time greats and I have been a fan of his since the 1950's. I went t
o this movie because Sean Connery was the main actor. I had not read revie
ws or had any prior knowledge of the movie. The movie surprised me quite a
bit. The scenery and sights were spectacular, but the plot was unreal to
the point of being ridiculous. In my mind this was not one of his better m
ovies it could be the worst. Why he chose to be in this movie is a mystery
. For me, going to this movie was a waste of my time. I will continue to g
o to his movies and add his movies to my video collection. But I can't see
wasting money to put this movie in my collection.

You don't have to do anything to get the update; it will be sent automatic
ally by Amazon. You may see the indicator light on your Echo will pulse bl
ue as the update is installed. The representative I spoke with said you ma
y get it more quickly by not using your Echo for several hours. I hit the
mute button on mine and received the update within a few hours. I'm very i
mpressed by Amazon's quick response to this issue!

Figure 13. Visualizing the value of the sentiment cell as it processes an IMDb reviews and an Amazon review. Red indicates negative sentiment while green indicates positive.

Based on this single sentiment neuron we can hopefully distinguish the sentiment polarity without any supervision of labelled data just by fitting a threshold to this single feature. We test this performance of this sentiment neuron some of previously used dataset, the results are presented in table 4.

Table 4. Comparison: polarity detection with supervised and unsupervised methods

Dataset	*Accuracy only with sentiment neuron*	*Accuracy with supervised learning*	*benchmark*
Stanford Sentiment Treebank	85.6%	**91.8** %	90.2%[35]
IMDb	91.8%	92.3%	**94.1% [36]**
MPQA2.0	76.2%	81.1%	**93.3%** [34]
Twitter2016	44.6%	54.7%	**65.1%** [37]
Twiiter2016 binary	75.9%	**84.5%**	----
OpinMind	90.5%	92.6%	**93.8% [29]**

We see for all the dataset we've tested, there is a drop of performance from supervised method to unsupervised method. However by using only the single sentiment neuron, we are still able to effectively classify some types of text like the IMDb movie reviews in which the accuracy only drops 0.6% compared with supervised method. But for datasets like Twitter2016 and MPQA2.0, depending only on the sentiment neuron will lead to a remarkable performance decrease around 10%. For these types of datasets the pretrained model based on Amazon reviews fails to represent the sentiment as effectively as it does for review text.

Nevertheless, there exist a compromise between the supervised learning and unsupervised learning. It is called weakly supervised learning which uses incomplete supervision where only a subset of training data are given with labels[31]. This compromise is especially useful when it is difficult to get strong supervision information like fully ground-truth labels due to the high cost of data labeling process, which is exactly the problem we are facing for our project.

Figure 14 shows the performance on Yelp dataset as a function of number of labelled training examples. The curve goes up sharply when the number of labelled samples increase from 0 to 10, then maintains at a high level while slowly increasing by only 1% across a 4 order of magnitude increase in training data. This characteristic can be very for weakly supervised learning, for example when we only have access to merely 10 labelled sample, our model still gives a good performance on polarity detection.

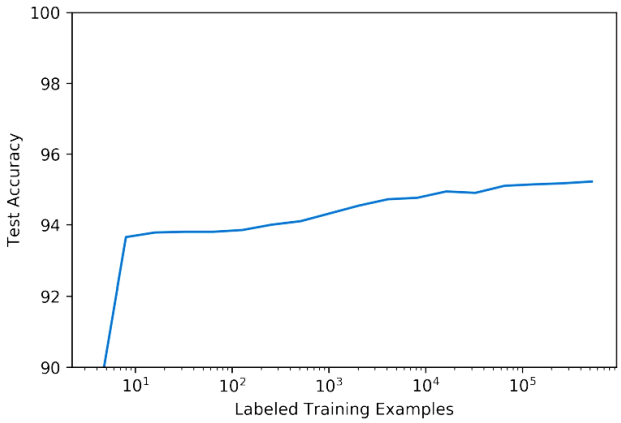


Figure 14. Test accuracy on Yelp dataset with varying number of labelled training examples. The model's performance plateaus after about ten labeled examples and only slow improves with additional data[15]

4.2.4 Model test for text generation

As stated in section 3.1, this model is originally designed for text prediction, so it can be certainly used as a generative model. Given the some initial text, the mLSTM model can naturally generate the following sequence of text which makes sense. Moreover we can customize the sentiment expressed in the generated text simply by setting the sentiment neuron to be positive or negative. From the samples of reviews we generate in figure 15, they are high quality and often include valid sentences.

```
This movie was totally disappointing
This movie was totally disappointing for me and I don't think it will ever be a cult cla
ssic.  It should not have been released and I am not sure it will ever be able to return
 to the streets of the streets of New York to try and recapture the spirit of the origin
al.  I was like this is a terrible movie but I have to say I would not want to be the sc
ene stealer in this film.  The only redeeming part was the scene in the doctor's ward.
The only thing that made me slightly sadden was the fact that the story of the film seem
ed to be that he was a hard worker, and that he was a good actor.  I think he was trying
 to make the movie look more real and then showing the hostage son that was killed by a
foreign gunman.  The film was so depressing that I could barely stand it.  I would not r
ecommend this film to anyone and I will not recommend it to anyone.  I am sorry I cannot
 recommend this film to anyone I know and I wish them anything but the best in life.  Th
is is a terrible movie and I would like to apologize to the people of
```

Figure 15. mLSTM used as generative model, a negative movie review generated by Amazon model by initializing with "*This movie was total disappointing*"

However it seems that the pretrained model based on Amazon reviews is constrained to handle only the product reviews. Due this limitation, we train a new one so that it can adapt to wider range of textual data. We select the CNN news corpus [22] as the training set. It contains the documents and accompanying questions from the news articles of CNN. There are approximately 90k documents, but we only select 11,469 of them to train the model, while 638 as the validation corpus.

The newly trained model is based on more sophisticated text, so it turns out to be able to handle wider range of topics like economy, politics, sciences etc., with more abundant vocabulary.

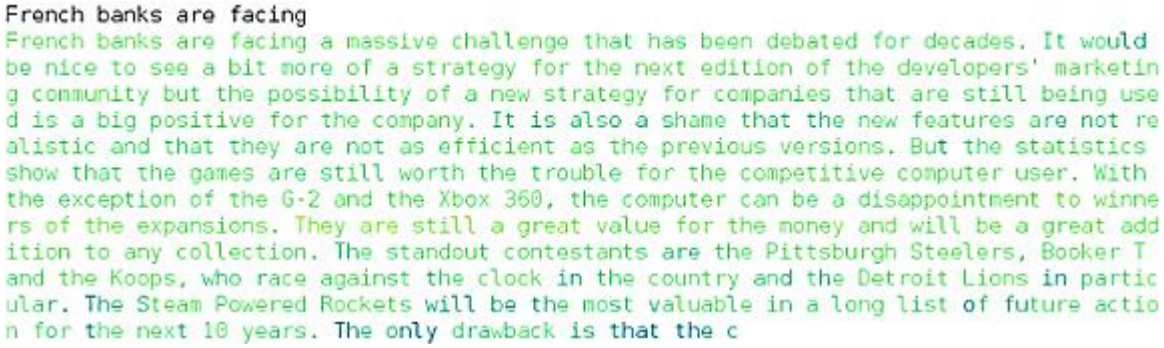


Figure 16. Example of the CNN model handling more complex topic, text initialized with "*French banks are facing*", fixed with positive sentiment

5 Future work

- **Stronger compatibility**

  Since the current model is only trained on relatively small scale of Bloomberg corpus, it is far from perfect for good sentiment representation. In next step we will try to train the model on larger dataset and other types of texts, see how the algorithm adapts to different nature of text.

- **Alternatives**

  There surely exist types of data that mLSTM model will fail to handle. In this case looking for the alternative unsupervised learning methods is necessary.

- **Adaptivity**

  So far we only deal with the polarity detection. In future work we expect to adapt the method to emotions analysis (emotion classification). This could be useful when the approach is applied to conversations like e-mails, chat and phone call transcripts.

Reference

[1] Liu, B., 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, *5*(1), pp.1-167.

[2] Das, S. and Chen, M., 2001, July. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific finance association annual conference (APFA)* (Vol. 35, p. 43).

[3] Morinaga, S., Yamanishi, K., Tateishi, K. and Fukushima, T., 2002, July. Mining product reputations on the web. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 341-349). ACM.

[4] Pang, B., Lee, L. and Vaithyanathan, S., 2002, July. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*(pp. 79-86). Association for Computational Linguistics.

[5] Turney, P.D., 2002, July. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 417-424). Association for Computational Linguistics.

[6] Wiebe, J., 2000. Learning subjective adjectives from corpora. *AAAI/IAAI*, *20*(0), p.0.

[7] Jin, X., Li, Y., Mah, T. and Tong, J., 2007, August. Sensitive webpage classification for content advertising. In *Proceedings of the 1st international workshop on Data mining and audience intelligence for advertising* (pp. 28-33). ACM.

[8] Hinton, G.E., Osindero, S. and Teh, Y.W., 2006. A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), pp.1527-1554.

[9] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. *Neural computation*, *9*(8), pp.1735-1780.

[10] Karpathy, A., Johnson, J. and Fei-Fei, L., 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

[11] Graves, A., 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

[12] Sutskever, I., Martens, J. and Hinton, G.E., 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (pp. 1017-1024).

[13] Krause, B., Lu, L., Murray, I. and Renals, S., 2016. Multiplicative LSTM for sequence modelling. *arXiv preprint arXiv:1609.07959*.

[14] Sutskever, I., Martens, J. and Hinton, G.E., 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (pp. 1017-1024).

[15] Radford, A., Jozefowicz, R. and Sutskever, I., 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

[16] Boden, M., 2002. A guide to recurrent neural networks and backpropagation. *the Dallas project*.

[17] McAuley, J., Pandey, R. and Leskovec, J., 2015, August. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). ACM.

[18] Pang, B. and Lee, L., 2005, June. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics* (pp. 115-124). Association for Computational Linguistics.

[19] 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. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1631-1642).

[20] Deng, L. and Wiebe, J., 2015. MPQA 3.0: An Entity/Event-Level Sentiment Corpus. In *HLT-NAACL* (pp. 1323-1328).

[21] Zhao, H., Lu, Z. and Poupart, P., 2015, July. Self-Adaptive Hierarchical Sentence Model. In *IJCAI* (pp. 4069-4076).

[22] Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M. and Blunsom, P., 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems* (pp. 1693-1701).

[23] Medhat, W., Hassan, A. and Korashy, H., 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, *5*(4), pp.1093-1113.

[24] Qiu, G., He, X., Zhang, F., Shi, Y., Bu, J. and Chen, C., 2010. DASA: dissatisfaction-oriented advertising based on sentiment analysis. *Expert Systems with Applications*, *37*(9), pp.6182-6191.

[25] Hatzivassiloglou, V. and McKeown, K.R., 1997, July. Predicting the semantic orientation of adjectives. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics* (pp. 174-181). Association for Computational Linguistics.

[26] Ko, Y. and Seo, J., 2000, July. Automatic text categorization by unsupervised learning. In *Proceedings of the 18th conference on Computational linguistics-Volume 1* (pp. 453-459). Association for Computational Linguistics.

[27] Xianghua, F., Guo, L., Yanyan, G. and Zhiqiang, W., 2013. Multi-aspect sentiment analysis for Chinese online social reviews based on topic modeling and HowNet lexicon. *Knowledge-Based Systems*, *37*, pp.186-195.

[28] Hu, M. and Liu, B., 2004, August. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 168-177). ACM.

[29] Strapparava, C. and Valitutti, A., 2004, May. Wordnet affect: an affective extension of wordnet. In *Lrec* (Vol. 4, pp. 1083-1086).

[30] Rosenthal, S., Farra, N. and Nakov, P., 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (pp. 502-518).

[31] Zhou, Z.H., 2017. A brief introduction to weakly supervised learning. *National Science Review*.

[32] Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y. and Potts, C., 2011, June. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (pp. 142-150). Association for Computational Linguistics.

[33] Zhang, X., Zhao, J. and LeCun, Y., 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems* (pp. 649-657).

[34] Zhao, H., Lu, Z. and Poupart, P., 2015, July. Self-Adaptive Hierarchical Sentence Model. In *IJCAI* (pp. 4069-4076).

[35] Looks, M., Herreshoff, M., Hutchins, D. and Norvig, P., 2017. Deep learning with dynamic computation graphs. *arXiv preprint arXiv:1702.02181*.

[36] Miyato, T., Dai, A.M. and Goodfellow, I., 2016. Adversarial Training Methods for Semi-Supervised Text Classification. *arXiv preprint arXiv:1605.07725*.

[37] Baziotis, C., Pelekis, N. and Doulkeridis, C., 2017. Datastories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (pp. 747-754).

[38] Baccianella, S., Esuli, A. and Sebastiani, F., 2010, May. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *LREC* (Vol. 10, No. 2010, pp. 2200-2204).

[39] Pak, A. and Paroubek, P., 2010, May. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc* (Vol. 10, No. 2010).