Stat NLP 2015

Project 1

News to Tweet

A method for compressing news articles to tweets

Pranav Dhar and Ismail Mustafa

Problem

Three International Space Station crew members landed in the snowy steppe to the northeast of the Kazakh city of Zhezkazgan on Friday, a NASA Television broadcast showed.

A capsule carrying NASA astronaut Kjell Lindgren, Russian cosmonaut Oleg Kononenko and Japan's Kimiya Yui was found by a search and rescue group whose four helicopters braved very strong winds and low clouds to reach the touchdown site.

Lindgren, Kononenko and Yui have been in orbit for nearly five months. Their replacements are slated to blast off from Kazakhstan's Baikonur Cosmodrome on Tuesday.

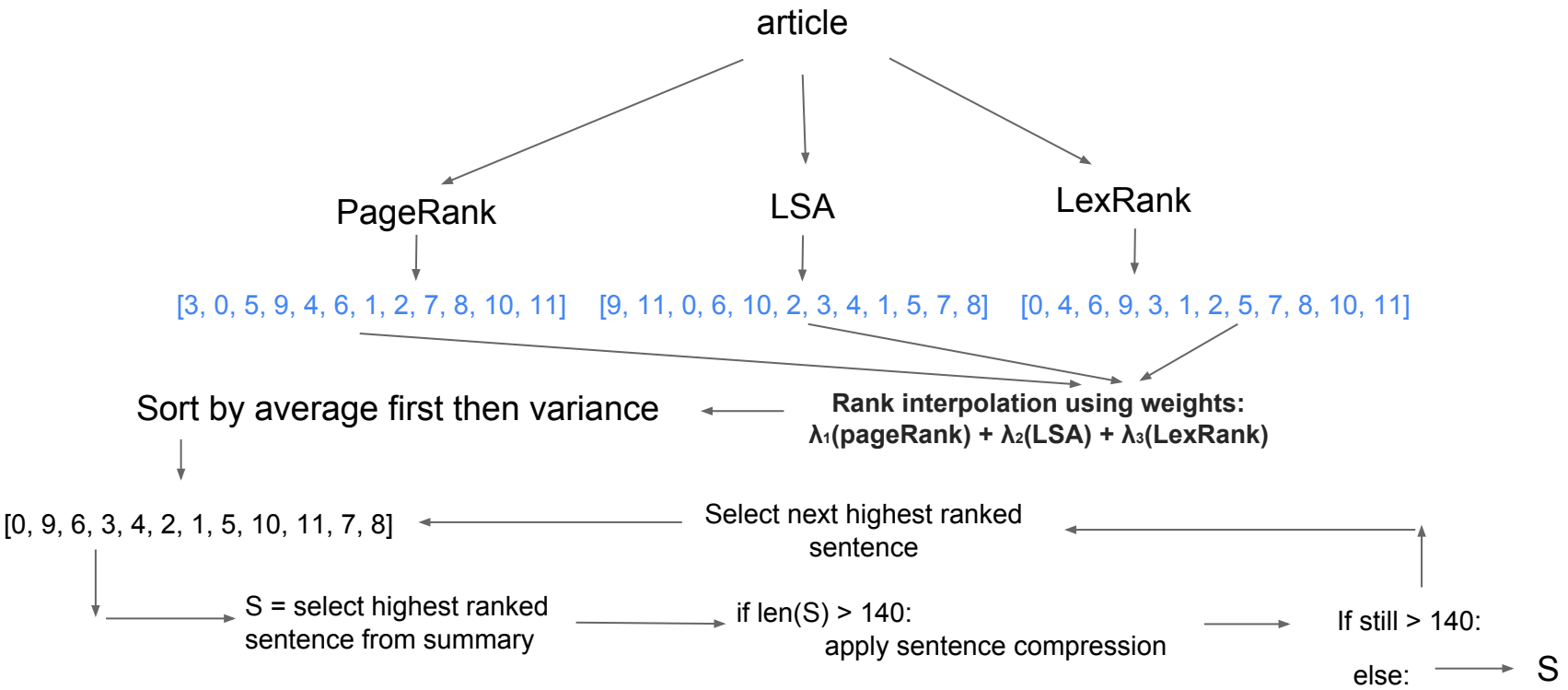
Left aboard the $100 billion station are U.S. astronaut and commander Scott Kelly and Russian flight engineers Mikhail Kornienko and Sergey Volkov.

Kelly and Kornienko are in the final months of a year-long mission, the longest stint in space since crews began living on the station in 2000. They are due to land on March 1.

Six Russian cosmonauts previously spent more than 300 days in space aboard the now-defunct Soviet station Mir. The longest flight lasted nearly 438 days.

Kelly and Kornienko's year-long mission is a trial run as the 15-nation station partnership begins planning for longer missions to the moon, Mars and other destinations beyond the space station.

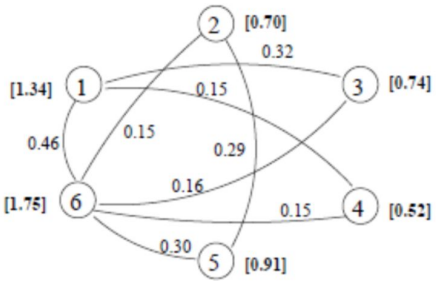
Approach


article
PageRank
LSA
LexRank
[3, 0, 5, 9, 4, 6, 1, 2, 7, 8, 10, 11]
[9, 11, 0, 6, 10, 2, 3, 4, 1, 5, 7, 8]
[0, 4, 6, 9, 3, 1, 2, 5, 7, 8, 10, 11]
Rank interpolation using weights:
λ1(pageRank) + λ2(LSA) + λ3(LexRank)
Sort by average first then variance
[0, 9, 6, 3, 4, 2, 1, 5, 10, 11, 7, 8]
Select next highest ranked sentence
S = select highest ranked sentence from summary
if len(S) > 140:
apply sentence compression
If still > 140:
else:
S

PageRank

$$tf_{i,j} = \frac{\text{freq}_{i,j}}{\max_l \text{freq}_{l,j}} \qquad isf_i = \log \frac{N}{n_i}$$

Weighted Undirected Graph



$$W(s_m, s_n) = \frac{\sum_{i=1}^{t} w_{i,m} \times w_{i,n}}{\sqrt{\sum_{i=1}^{t} w_{i,m}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,n}^2}}$$

$$PR(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{PR(V_j)}{|Out(V_j)|}$$

where d is a parameter set between 0 and 1.

"Graph Based Algorithm For Text Summarization." (n.d.): n. pag. Web. https://kenai.com/downloads/textsummarizer/FinalWriteups.pdf

LexRank

$$\text{idf-modified-cosine}(x, y) = \frac{\sum_{w \in x,y} \text{tf}_{w,x}\text{tf}_{w,y}(\text{idf}_w)^2}{\sqrt{\sum_{x_i \in x}(\text{tf}_{x_i,x}\text{idf}_{x_i})^2} \times \sqrt{\sum_{y_i \in y}(\text{tf}_{y_i,y}\text{idf}_{y_i})^2}}$$

	1	2	3	4	5	6	7	8	9	10	11
1	1.00	0.45	0.02	0.17	0.03	0.22	0.03	0.28	0.06	0.06	0.00
2	0.45	1.00	0.16	0.27	0.03	0.19	0.03	0.21	0.03	0.15	0.00
3	0.02	0.16	1.00	0.03	0.00	0.01	0.03	0.04	0.00	0.01	0.00
4	0.17	0.27	0.03	1.00	0.01	0.16	0.28	0.17	0.00	0.09	0.01
5	0.03	0.03	0.00	0.01	1.00	0.29	0.05	0.15	0.20	0.04	0.18
6	0.22	0.19	0.01	0.16	0.29	1.00	0.05	0.29	0.04	0.20	0.03
7	0.03	0.03	0.03	0.28	0.05	0.05	1.00	0.06	0.00	0.00	0.01
8	0.28	0.21	0.04	0.17	0.15	0.29	0.06	1.00	0.25	0.20	0.17
9	0.06	0.03	0.00	0.00	0.20	0.04	0.00	0.25	1.00	0.26	0.38
10	0.06	0.15	0.01	0.09	0.04	0.20	0.00	0.20	0.26	1.00	0.12
11	0.00	0.00	0.00	0.01	0.18	0.03	0.01	0.17	0.38	0.12	1.00

input : A stochastic, irreducible and aperiodic matrix $\mathbf{M}$
input : matrix size N, error tolerance ϵ
output: eigenvector $\mathbf{p}$

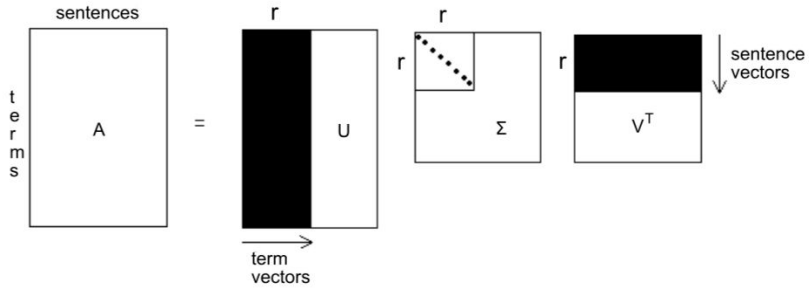
$\mathbf{p}_0 = \frac{1}{N}\mathbf{1}$;
t=0;
repeat
    t=t+1;
    $\mathbf{p}_t = \mathbf{M}^T\mathbf{p}_{t-1}$;
    $\delta = ||\mathbf{p}_t - \mathbf{p}_{t-1}||$;
until $\delta < \epsilon$;
return $\mathbf{p}_t$;

Erkan, Gunes, and Dragomir R. Radev. "LexRank: Graph-based Centrality as Salience in Text Summarization." *Journal of Artificial Intelligence Research* 22 (2004). Web. 1 Nov. 2015.

LSA

Words that are close in meaning will occur in similar pieces of text.

$$tf_{i,j} = \frac{\text{freq}_{i,j}}{\max_l \text{freq}_{l,j}}$$



$$B = \Sigma^2 \cdot V^T$$

$$s_k = \sqrt{\sum_{i=1}^{r} b_{i,k}^2}, \qquad B = \begin{pmatrix} v_{1,1}\sigma_1^2 & v_{1,2}\sigma_1^2 & \dots & v_{1,n}\sigma_1^2 \\ v_{2,1}\sigma_2^2 & v_{2,2}\sigma_2^2 & \dots & v_{2,n}\sigma_2^2 \\ \dots & \dots & \dots & \dots \\ v_{r,1}\sigma_r^2 & v_{r,2}\sigma_r^2 & \dots & v_{r,n}\sigma_r^2 \end{pmatrix}.$$

Steinberger, Josef. *Text Summarization within the LSA Framework*. Thesis. University of West Bohemia, 2007. N.p.: n.p., n.d. Print.

Sentence Compression

[S | S <- (all possible ways to split sentence)]

PPDB: The Paraphrase Database

len(target) < len(source)

[P | P <- (all possible paraphrases)]

Calculate trigram score for each paraphrased sentence using trigram language model with kneser ney smoothing.

Pick sentence that fits language model the best (this model is very conservative)

http://www.cis.upenn.edu/~ccb/ppdb/

Evaluation

Corpus: DUC 2002

- NIST Document understanding conference
- Convenient for comparison (used by several papers we reference)

Metric:

- Rouge (Recall-Oriented Understudy for Gisting Evaluation)
  - n-gram, word sequences, word pairs
    - Recall
    - Precision
    - F1 score = 2*(precision * recall)/(precision + recall)

Related work/baselines

PageRank

- "Graph Based Algorithm For Text Summarization." (n.d.): n. pag. Web. https://kenai.com/downloads/textsummarizer/FinalWriteups.pdf
- Precision: 0.597, Recall: 0.206, F1 score: 0.306

LexRank

- Erkan, Gunes, and Dragomir R. Radev. "LexRank: Graph-based Centrality as Salience in Text Summarization." *Journal of Artificial Intelligence Research* 22 (2004). Web. 1 Nov. 2015.
- F1 score: 0.3582 - 0.4443 (topics)

LSA

- Steinberger, Josef. *Text Summarization within the LSA Framework*. Thesis. University of West Bohemia, 2007. N.p.: n.p., n.d. Print.
- F1 Score: 0.42776

Results

Algorithm	Precision	Recall	F1 Score
LSA (r=5)	0.463	0.539	0.474
PageRank (d=0.5)	0.464	0.349	0.394
LexRank (t= 0.01)	0.515	0.226	0.305
Rank Interpolation λ_1=0.1 λ_2=0.8 λ_3=0.1	0.387	0.531	0.445

Questions/Demo

http://3d4d7ebd.ngrok.com/

Project 2

Build a Real-time Emoji Recommendation System

Boyang Zheng and Yi Liu

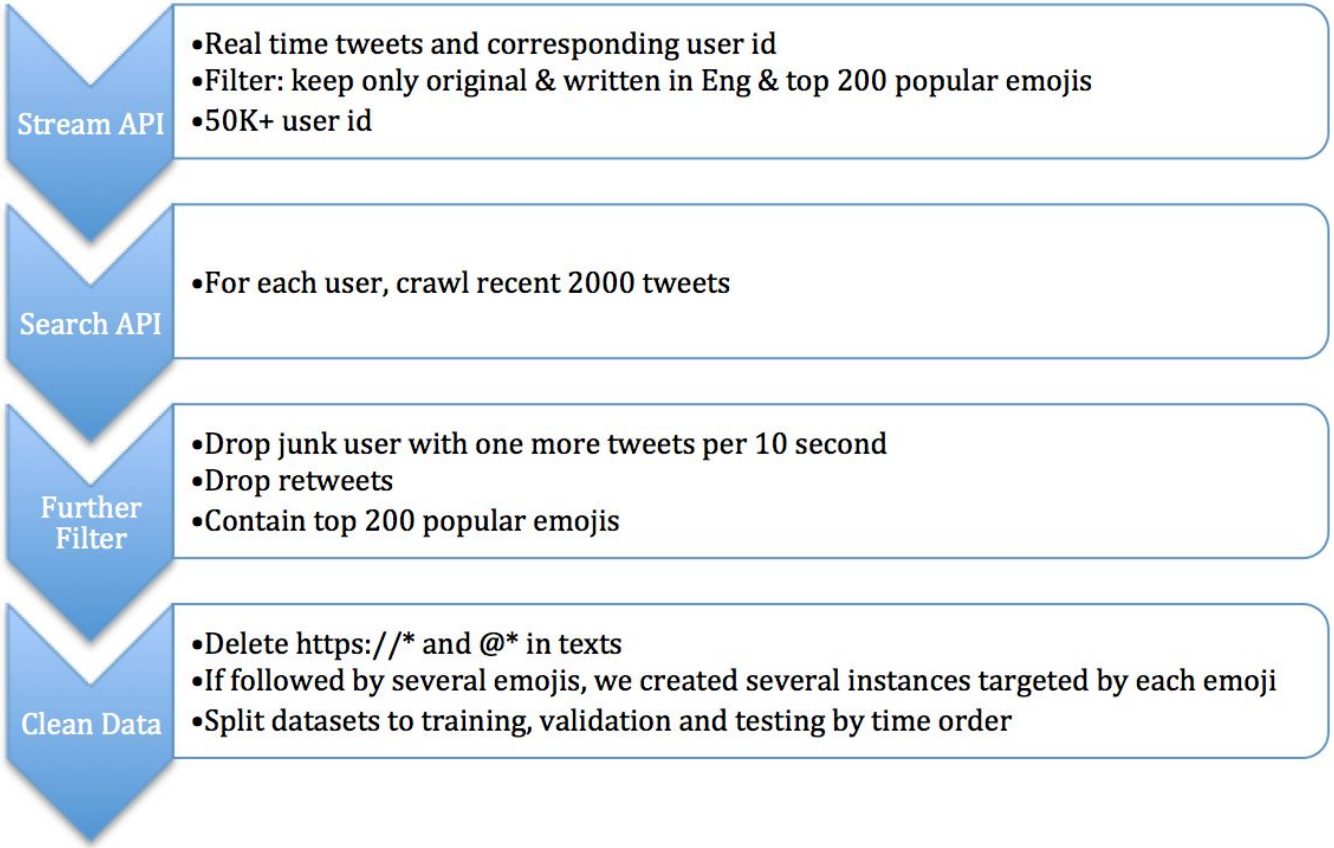
The Motivations

- Emoji is very popular. (92% online user using emojis).
- More than 500 emojis in total, it’s hard to find the right one.
- Lack of emoji recommendation applications.

The Goals

- Understand the meaning of emojis.
- Make predictions on emojis based on texts.
- Build an emoji recommendation engine.

Data Collection & Data Preprocessing



Emoji Understanding

- ## Term Frequency & TFIDF

 king, princess, blonde, cinderella, queen, blondes, princesses, crown, prinny

 marry, engagement, married, ring, proposed, wedding, wife, please, rings

 stack, payday, cash, rich, money, bankroll, selling, paycheck, grind, benjamins

- ## Topic Model (Latent Dirichlet Allocation)

  - ***merry, christmas,*** perfectly, ***happy, birthday,*** ily, dance, amo, reunit, sunshine, ***hbd,*** miss, ***18th, present,*** heart, happiest, best, bday, tree, excited, cake, bestfriend, sister, 21st. 🎈💌🎂🎄🎊
  - ***sunshine,*** pizza, ***sun, sunny, xx, summer, tan, beach,*** weather, ***pool, shine, florida,*** joy, legend, degree, adore, rise, universe, holiday, spring, burnt, relax, lake, proud, congrat, ***california, heat, vacation,*** hungry, outside, bbq, weekend, garden, workout, warm, ray, adventure, ***miami,*** swim, cheer 😎🌞🍀🍕
  - ***puppy, cat, kitty***, idol, africa, travel, ***monkey, dog***, light, ***pussy,*** matt, ***pet,*** artist, ***cuddly,*** loud, freaking, calumny, bruh, ear, ***cutest, animal, earth, cute,*** biggest, ***dawg*** 🐶🌍🌿🐱

Emoji Understanding

- **Emotion Analysis**

NRC Word-Emotion Association Lexicon (14182 unigrams; 0-1 score; 10 emotions)

- **Network Analysis**

- Group tweets using the same emojis.
- Treat word list for each emoji as a document
- Calculate document similarity.
- Visualize using the distance matrix.

Model Preparation

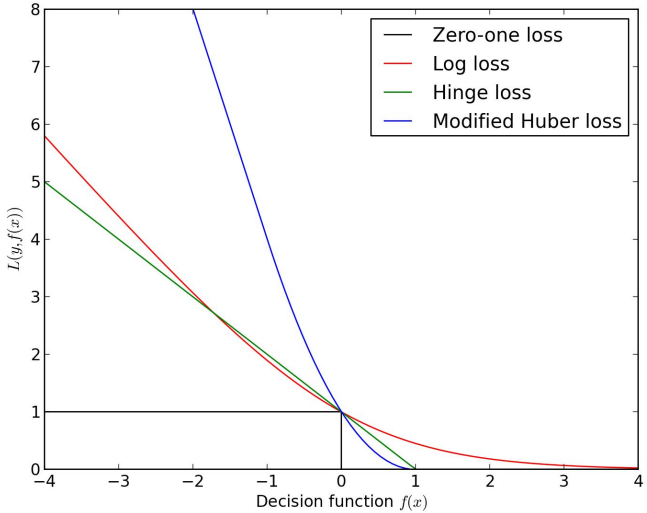
- **Texts Preprocessing**
  - Clean texts: drop stopwords, punctuations.
  - Stem words: tense, plural, singular.
  - Build lexicon: set minimum word frequency and user frequency.

- **Potential Features**
  - Word frequency/TF-idf (unigram, bigrams).
  - Topic distribution.
  - Emotion Score.

Classification

Our target is to predict which emoji the user is going to use while they are typing. We regard this as a ***classification problem*** in which to classify top 200 emojis based on texts. Since the dataset is large (~10M instances), we applied ***stochastic gradient descent*** (SGD), which is an approach of discriminative learning of linear classifier with convex loss.

- **Baseline**
  - Recommend the most popular emoji.
- **Stochastic Gradient Descent Classifier**
  - Logistic loss (Logistic regression).
  - Hinge loss (Support vector machine).
  - Modified huber loss.
- **Experiments**
  - Unbalanced dataset: adjusted weight.
  - Regularization: l1/l2; tuned the penalty constant alpha.
  - Different combinations of potential features.



Evaluation

- **Accuracy**
- **Similarity Score**
  - Motivation:
    - Some emojis are similar. Recommending similar emojis is acceptable.
    - Penalizing all incorrect prediction equally is not fair.
  - Approach:
    - Cosine similarity between word frequency vectors of two emojis.

	Training Set		Validation Set	
Model	Accuracy	Similarity	Accuracy	Similarity
Baseline	0.1396	0.7556	0.1396	0.7538
Logistic Regression	0.2693	0.8127	0.2543	0.8051
SVM	0.2251	0.7622	0.2108	0.7560
Modified Huber	0.2863	0.8141	0.2633	0.8033

Evaluation

- **User Judgement**
  - A toy recommendation engine.
    - Take user's input as texts; output the predicted emoji.

Texts	True	Rec.
I miss you	😔	😭
Dream your DREAMS	🙏	😴
hate seeing my mom upset	😭	😂
I need to figure out what I wanna do with me life	😹	😩
happy 17th birthday my loveee	♥	🎉
I need to rethink my life	😂	😂

```
Input? happy birthday
🎉
Input? I miss you
😭
Input? good morning
😊
Input? good night
❤
Input? I had a sweet dream
😍
Input? I need a cup of coffee
☕
Input? thank you
😘
Input? pray for Paris
🙏
Input?
```

Further: Recurrent Neural Network with LSTM

Previous results are based on bag of words model, which didn’t take the order of word sequence into consideration. But the order may be useful, so we are going to use LSTM to capture the order information and apply to this classification task.

Conclusion

1. Understanding emoji: We found close **relationship between emojis and texts** (word frequency for unigram/bigram, topic distribution and emotions). We used them as our features. Inspired by the network visualization, we used similarity as one of our evaluation metrics.

2. We regarded this emoji recommendation task **as a multi-classification problem**. We used stochastic gradient descent classifier. Finally, we achieved **26% accuracy and 0.8 similarity score** on one-emoji prediction. Based on this, we built a **toy emoji recommendation engine**.

3. Further: we are going to use the **LSTM** which consider the term sequence order.

Thank you !

Native [1]	Apple [2]	Android [3]	Android [3]	Symbola [4]	Twitter [5]	Phantom [6]	Unicode	Bytes (UTF-8)	Description
							U+1F601	\xF0\x9F\x98\x81	grinning face with smiling eyes
							U+1F602	\xF0\x9F\x98\x82	face with tears of joy
							U+1F603	\xF0\x9F\x98\x83	smiling face with open mouth
							U+1F604	\xF0\x9F\x98\x84	smiling face with open mouth and smiling eyes
							U+1F605	\xF0\x9F\x98\x85	smiling face with open mouth and cold sweat
							U+1F606	\xF0\x9F\x98\x86	smiling face with open mouth and tightly-closed eyes
							U+1F609	\xF0\x9F\x98\x89	winking face
							U+1F60A	\xF0\x9F\x98\x8A	smiling face with smiling eyes

Project 3

Statistical Natural Language Processing - Fall 2015

Courant Institute of Mathematical Sciences

Analytics Project: Video to Text Description

Team: Saurabh Pujar, Ashish Yadav

Problem Setting:

•A large amount of data is stored in media like videos and images. Image descriptions do not capture temporal data.

•Solving the visual symbol grounding problem has long been a goal of artificial intelligence.

•In order to search videos we rely on tags. Tags do not always capture all the information.

•Video descriptions will facilitate information extraction

*

Related Work:

The inspiration for this project is the work of Prof. Raymond Mooney and others at the Machine
Learning Research group in University of Texas at Austin.

Evaluation metric used is METEOR. Other metrics like Bleu are also referenced

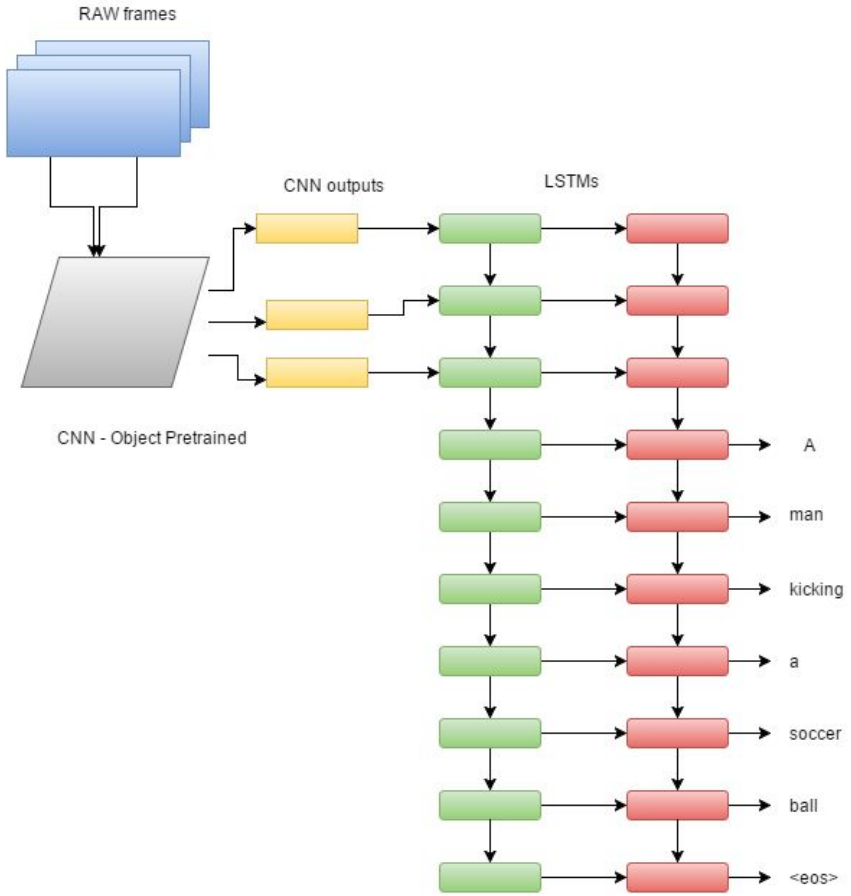
Data Set:

- We use Microsoft Research Video Description Corpus.

- This data consists of about 120K sentences collected during the summer of 2010.

- The result is a set of roughly parallel descriptions of more than 2,000 video snippets.

Design:


RAW frames
CNN outputs
LSTMs
CNN - Object Pretrained
A
man
kicking
a
soccer
ball
<eos>

*

Infrastructure:

CNN, LSTM: Caffe, HPC cluster

Evaluation: Python, NLTK library, Word2Vec

Experiments:

1. Alter the number of frames

2. Generate many sentences for the same video and choose the best match.

3. Generate many sentences for the same video and summarize.

Evaluation Metric 1: Baseline

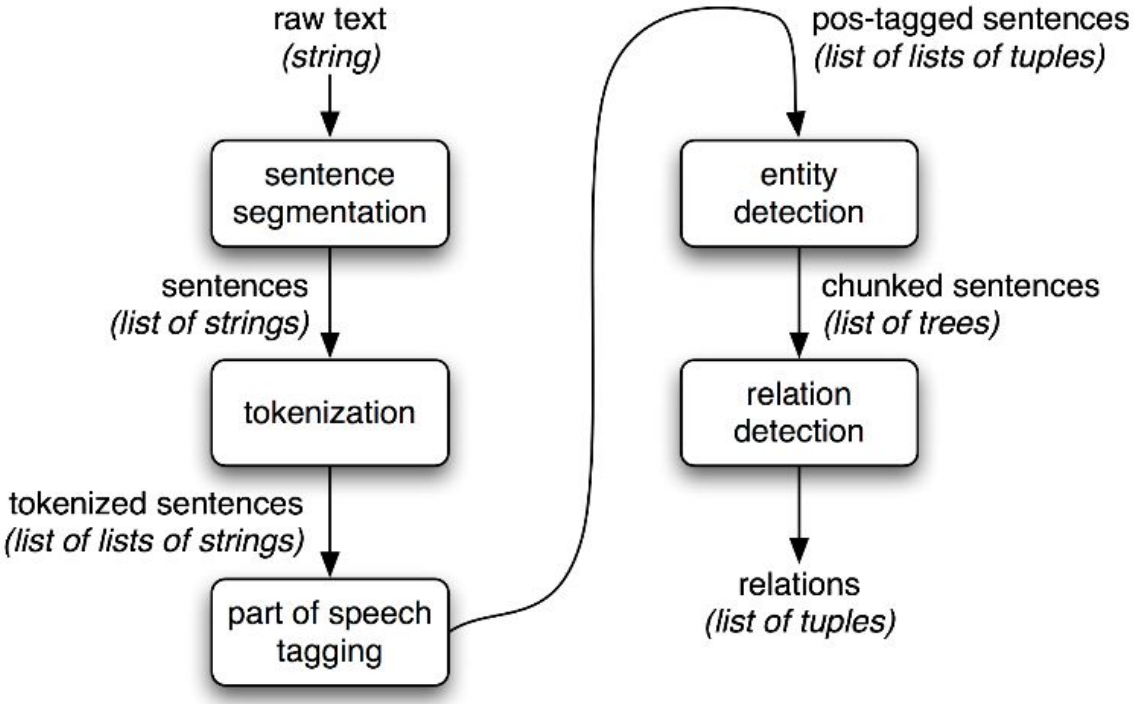
- Words that match precisely with the words in the description.
- The number of words matching for a sentence will give us a percentage.
- Use this to get an accuracy for the whole corpus.

*

Evaluation Metric 2: Part of speech comparison

- Figuring out part of speech tags and named entities in the sentence then comparing parts of speech and named entities.
- Different priority will be assigned to different parts of speech.
- WordNet/Word2Vec to look for synonyms

Parts of Speech:



Performance Metric 3: SOV triples

•Obtaining <subject, object, verb> triples from sentences and checking for only these three words.

•This will be done by Stanford NLP dependency parsing

•Stanford NLP parser is integrated with NLTK Library

*

Preliminary Results:

We have trained our model in the training data. We will now run experiments and test the output.

References:

[1] Sequence to Sequence – Video to Text. Raymond Mooney et.al

[2] Translating Videos to Natural Language Using Deep Recurrent Neural Networks. Raymond Mooney et. Al

[3] http://www.nltk.org/book/ch07.html

Thank you!

*

Project 4

POS Tagging and Chunking with Subword2Word models

Presentation

Project 5

Predicting Ratings From Yelp Reviews

Statistical Natural Language Processing

Instructor: Slav Petrov

Abhishek Shah

Rama Krishna Raju

Yelp Dataset Challenge

- Motivation: Glancing rating vs. reading a review
- Problem: Traditional Multiclass Classification
  - Input: Review
  - Output: Rating

Reviews !!!

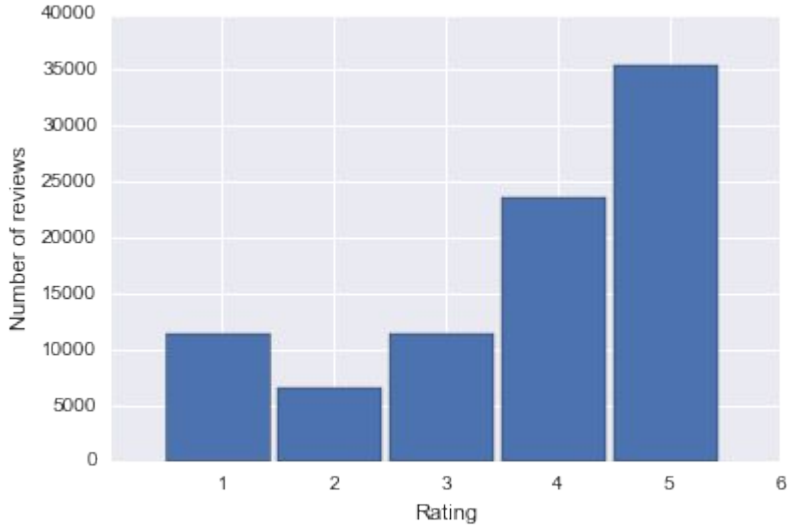
- Big fan of Ian. And his pizza.
- Best sushi in Pittsburgh.
- Quite a disappointment for us.
- Over rated
- Food is good. But roach crawling on wall but disgusting. Service can be improved

Prior Work

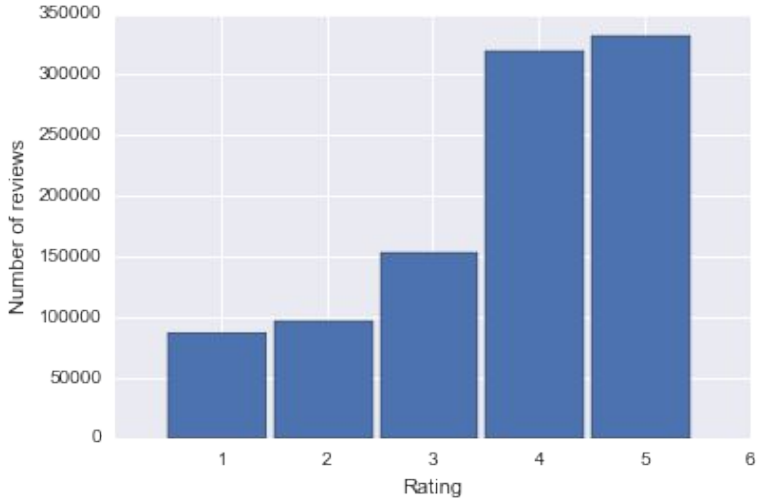
- Sentiment Analysis
  - Infer sentiment polarity: positive vs. negative
- Aspect based sentiment
  - Sentiment in a little more details
  - Topic models were found to be useful
- Rating prediction
  - Semantic topic modeling approach

Yelp DataSet

- Shopping Category



- Restaurant Category



Baseline Model and Evaluation Methods

- Baseline model
  - Simple heuristic model: Average of all the ratings

- Evaluation methods in prior work
  - Accuracy of the classifier
  - Perplexity
  - Root Mean Square Error

- We choose Root Mean Square Error

Approaches

- Features
  - Bag of words – TfIdf
  - TfIdf after stopword removal
  - POS tags
  - Review statistics like – average length, number of sentences.

- ML based classification
  - Logistic regression
  - Multinomial Naive Bayes
  - SVM

Other features

- LDA
  - Featured reviews as a linear combination of topic models
  - Used these features for both train & test
    - Did not work.
    - It was used in prior work for aspect based sentiment analysis.

- Sentiment
  - Rating prediction is a special case of sentiment prediction
  - Trained a sentiment classifier
  - Rating classifier + sentiment classifier
    - It improved.
    - To try, if LDA features helps in sentiment prediction.

Results

Models	Root Mean Square Error
Baseline	1.58
Adjectives	1.33
BOW (tfidf)	1.02
BOW + Length	1.08
BOW + Sentiment Classifier	0.95

Confusion Matrix	1	2	3	4	5
1	556	146	164	182	94
2	94	120	205	178	66
3	28	90	392	486	139
4	13	56	318	1161	809
5	23	27	180	891	2415

Classifying across the categories

Most similar words of “good”
great
decent
terrific
superb

Take words which are the closest to the word obtained from word2vec and add in to the unigram corpus of the test dataset of other category test. For Example,

> Best pizza. Number #1 place I take friends from out of town. Steak and fries & bbq chicken and bacon are my favs! Super awesome!

Buy Buy's Buys best cheese yogurt salad frozen SCAMS keg preview 60k 4 2 3 98 store shop spot FABO i Valentina Lesco furious recover persevere recreate straighten girlfriends relatives family MIL refrain utilize 3k Thus cashing periodically aggravated mindset payoff skateboard eliminated refresher Charlotte Madison valley Phx Smashburger Maggianos Grimaldi's Balboa Also ensuring immensely confronted creamy sundae moist buns luscious saucers stopper sprinkles cheese salad pizza rotisserie Also ensuring immensely confronted almond frosting gummy tortilla were aren't They're they're My our hers wife's Tiffanys affiliates demerit eReaders super Very duper SUPER amazing incredible great fantastic

Conclusions

- Future work
  - Capturing semantic rating space of users
  - Paragraph Vector instead of averaging word vectors

- Questions ?

References

- Jong, Jason, Predicting Rating with Sentiment Analysis (2011)
- Titov, Ivan et al., A Joint Model of Text and Aspect Ratings for Sentiment Summarization(2008)
- Qu, Lizhen et al. The Bag of Opinions Method for Review Rating Prediction from Sparse Text Patterns(2010)
- Tang, Duyu et. al, Learning Semantic Representations of Users and Products for Document Level Sentiment Classification
- Zhang, Yinshi, Semantic Feature Analysis and Mining for Yelp Rating Prediction

Project 6

Binary Tweet Sentiment Quantification

Runpeng Chen, Aaron McKinstry, Jonay Trénous

Quantification vs Classification

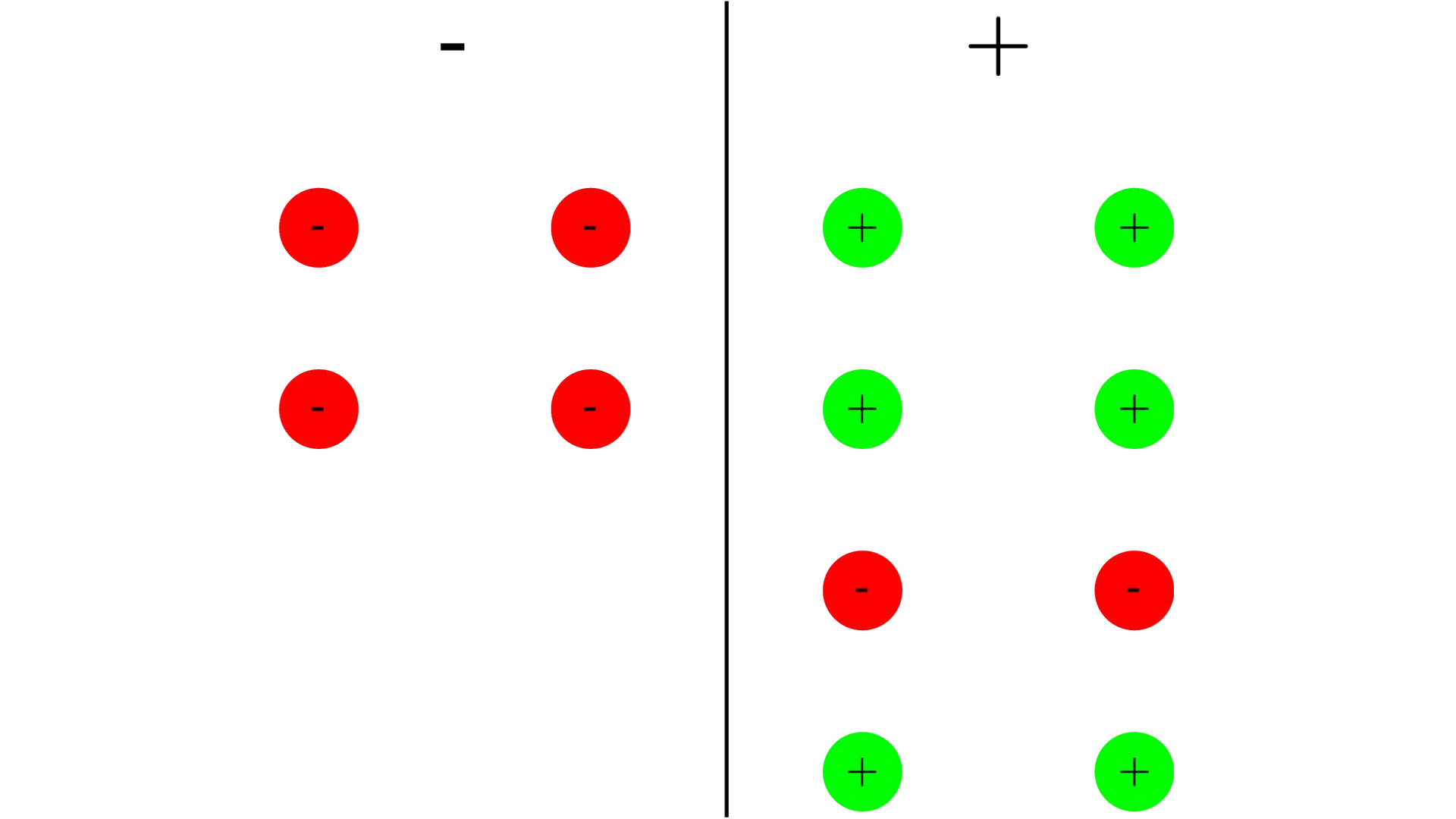
Classification:

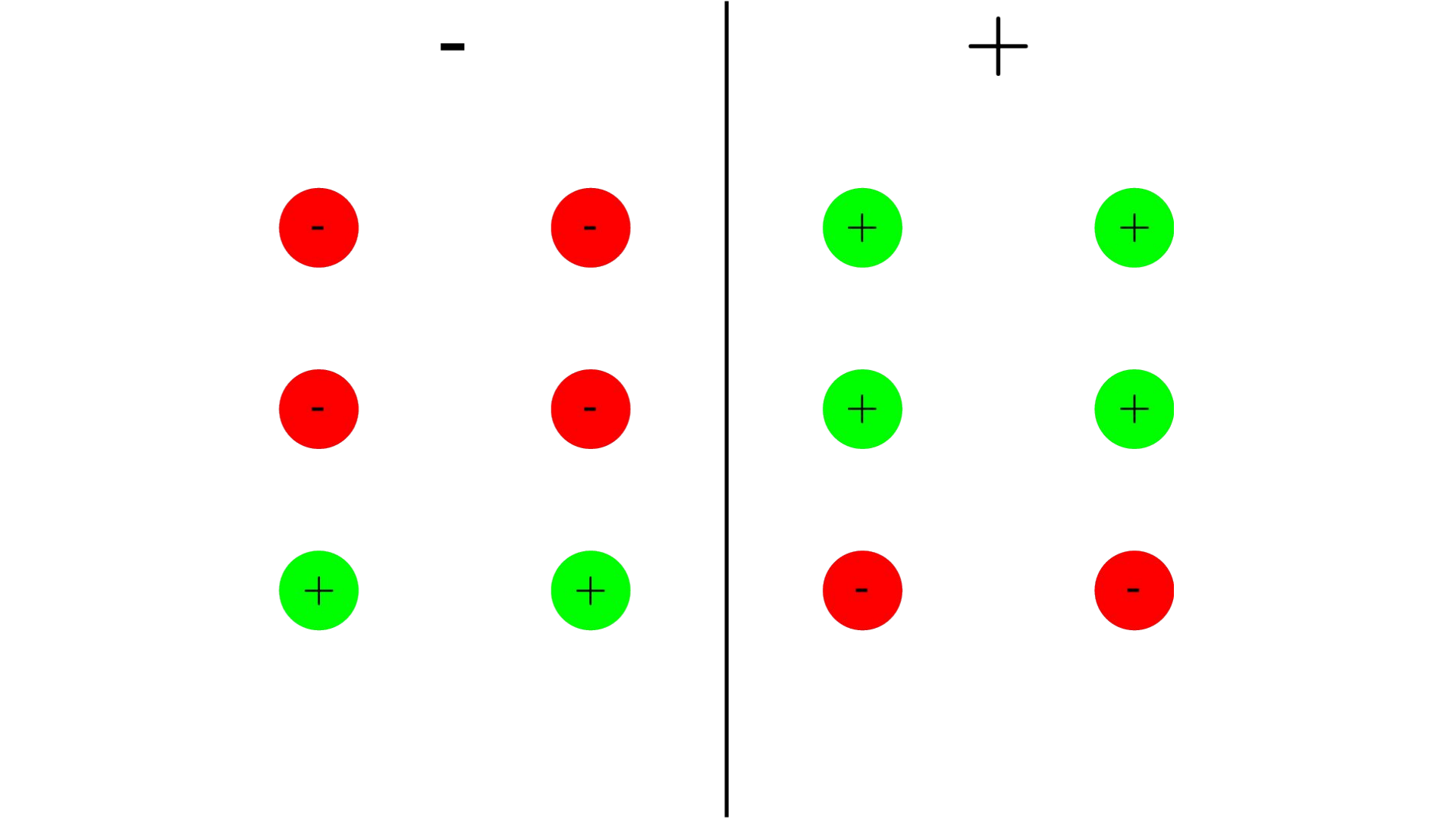
$$x \rightarrow y \in \{-1, +1\}$$

Quantification vs Classification

Quantification:

$$S = \{d_1, ..., d_n\} \rightarrow \mathbf{P}(\text{positive} \mid S)$$





Kullback Leibler Divergence

$$\mathrm{KLD(q,p)} = p(+) \log \frac{p(+)}{q(+)} + p(-) \log \frac{p(-)}{q(-)}$$

Explicit Loss Minimization

- Structured SVMs (Esuli and Sebastiani, 2014)

Explicit Loss Minimization

- Our Approach: Neural Net Based ELM

Data

Data: Human anotation

100 topics XXX tweets

```
amazon prime    negative    Trying to watch Any Given Sunday on Amazon Prime and there's big chunks of the film cut out
amazon prime    positive    Good to see @eamonngriffin new book available in hard copy.  Amazon Prime bringing it tomorrow.
amazon prime    positive    On the ball, @Amazon: 2 emails today announcing that Clarkson, Hammond and May are to make a
series about cars for Amazon Prime! Who knew?
amazon prime    positive    I have discovered that Bobby's World has unlimited streaming on Amazon Prime. I may or may not be
watching it while I'm on the bike.
amazon prime    negative    Heads up: An important change to Amazon Prime effective August 1 will limit how you can share
free P... http://t.co/BndStpoVJq @debtblag
amazon prime    negative    I'm bummed with Amazon Prime right now. I ordered something and it was supposed to be delivered
today, but it wasn't and I leave tomorrow :(
amazon prime    positive    Hammond, Clarkson and May have signed with Amazon Prime http://t.co/MMxlghe3Ja via
@wordpressdotcom #AmazonPrime #Hammond #Clarkson #May
amazon prime    positive    @DailyJulianne But if I need a power adaptor that may be my next Amazon Prime buy tomorrow...tbh
the luggage alone was stressful enough.
```

Baseline

For this question, for each sentence, we consider it as the Text Categorization Problem.

We choose Maximum Entropy Model as Baseline.

1. Input: We take the positive and negative as the label. For each word in the sentence, we choose them as the feature.
2. We took the score as probabilities:

$$P(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{\exp(\mathbf{w}^\top \mathbf{f}(\mathbf{y}))}{\sum_{\mathbf{y}'} \exp(\mathbf{w}^\top \mathbf{f}(\mathbf{y}'))}$$

3. Maximize the (log) conditional likelihood of training data

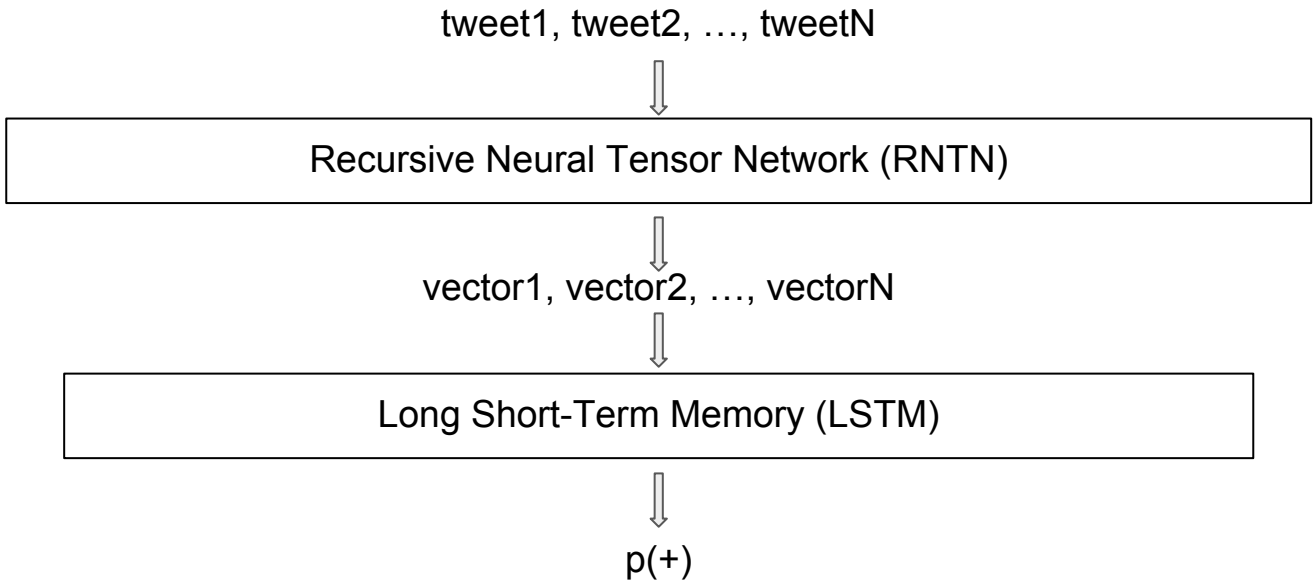
Testdata

1. Treat each sentence individually and do the classification. If the probability of positive is larger than negative, we consider it as positive. And then we count the positives to calculate the proportion.

Result

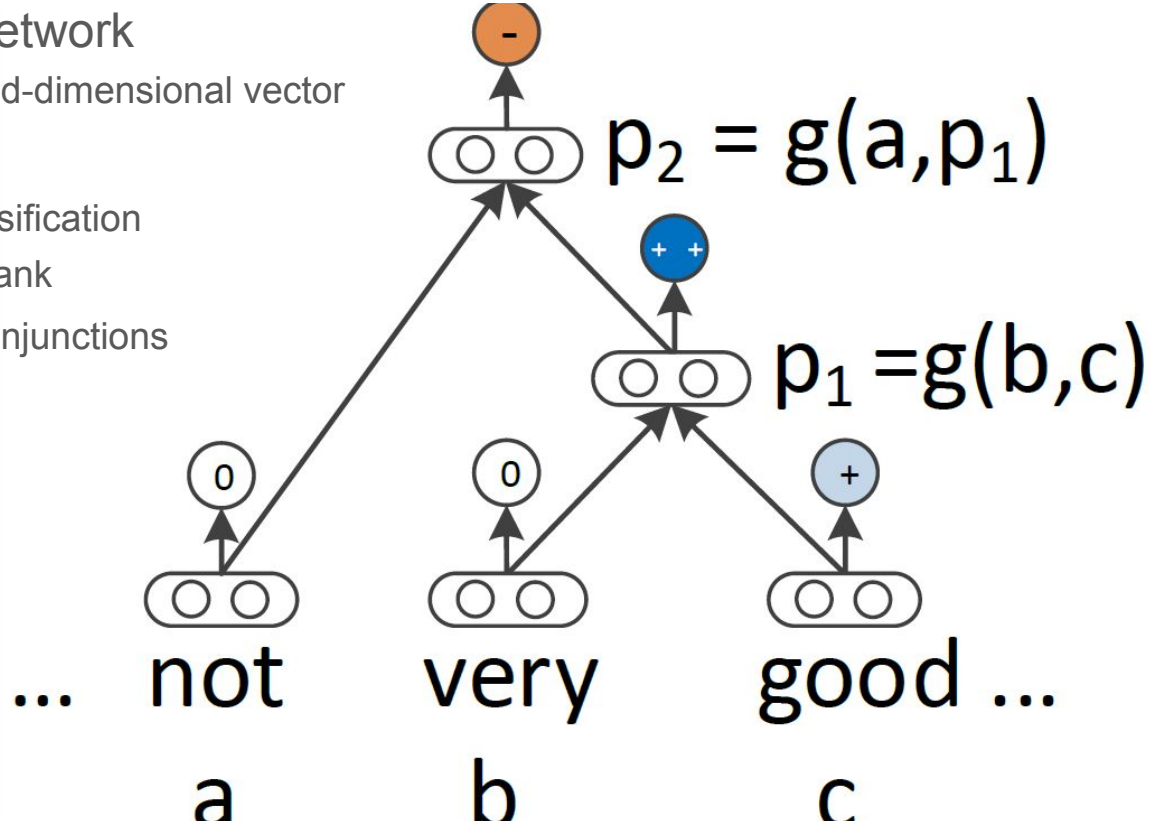
1. Score 0.379-->0.187

LSTM Model


tweet1, tweet2, …, tweetN
Recursive Neural Tensor Network (RNTN)
vector1, vector2, …, vectorN
Long Short-Term Memory (LSTM)
p(+)

Recursive Neural Tensor Network

- Recurrent Neural Tensor Network
  - (sentence, parse tree) => fixed-dimensional vector
- Our RNTN:
  - pre-trained for sentiment classification on Stanford Sentiment Treebank
  - able to capture contrastive conjunctions and negations



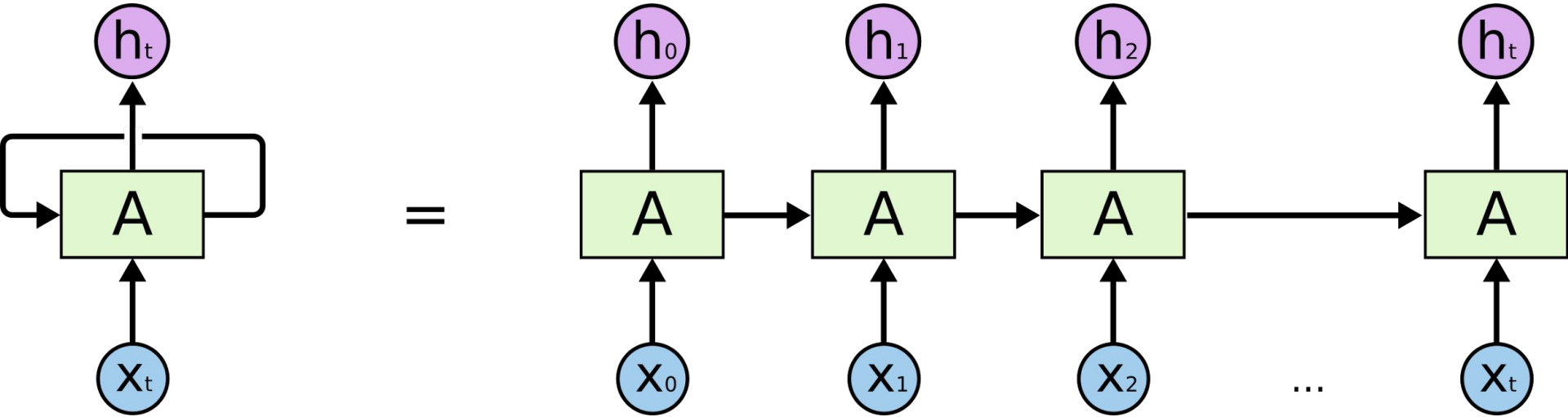
Richard Socher, Alex Perelygin, Jean Wu,Jason Chuang, Christopher Manning,Andrew Ng and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)

Long Short-Term Memory

- Neural network
  - handles arbitrary numbers of inputs
  - good empirical results
    - able to learn time-lagged dependencies in data (100’s of steps)

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (November 1997), 1735-1780. DOI=http://dx.doi.org/10.1162/neco.1997.9.8.1735

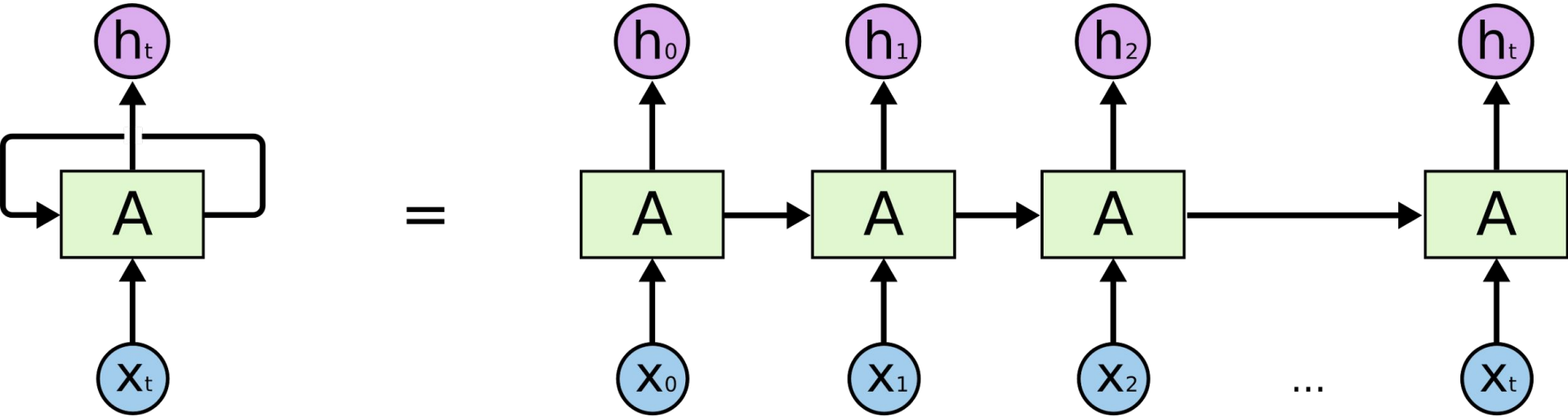
Long Short-Term Memory



Colah's blog (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

Long Short-Term Memory

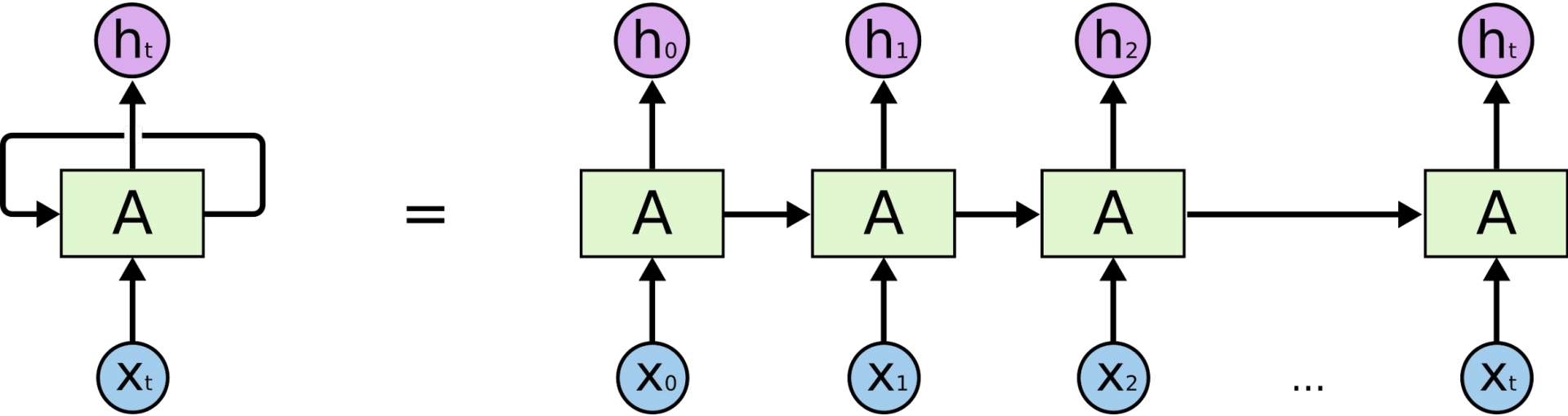
- In our case:
  - x(i) is the tweet vector for tweet (i)
  - h(i) is the fraction of positive tweets among x(0),...,x(i)



Colah's blog (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

Long Short-Term Memory

- We can directly optimize the KLD divergence

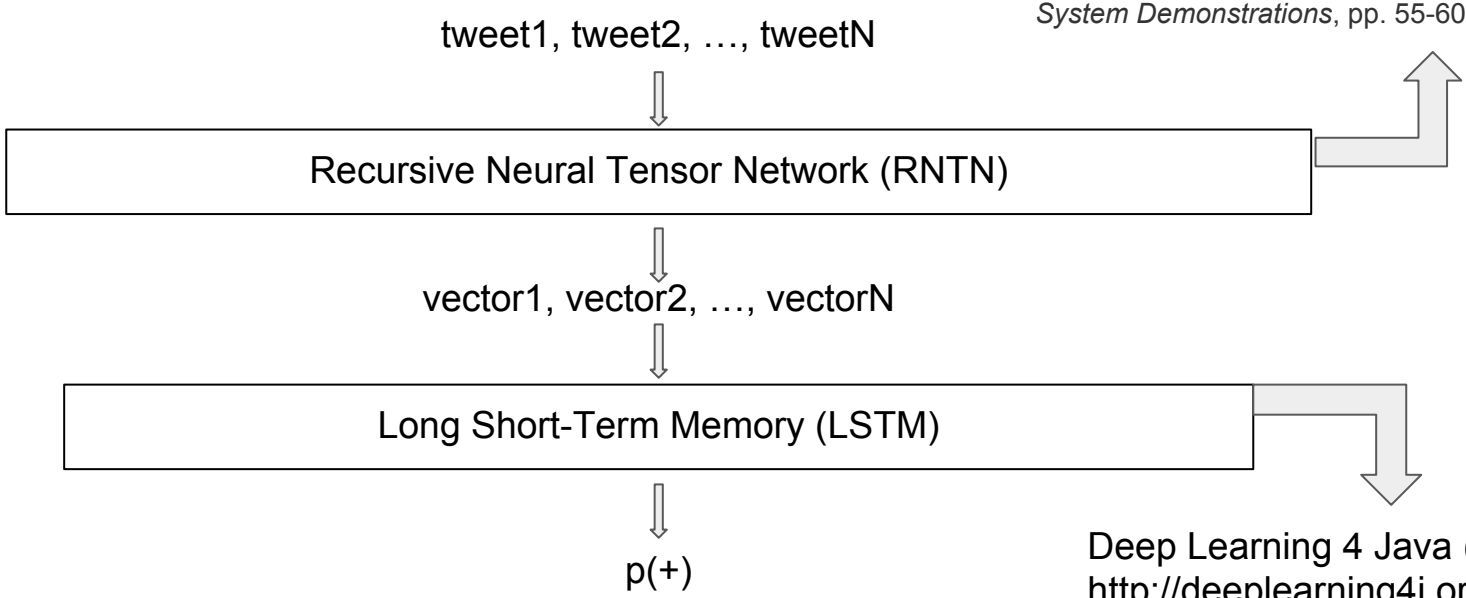


Colah's blog (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

LSTM Model

Stanford CoreNLP Java Library

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55-60.



PROJECT 7

Text Classification of Reddit Posts

Jackie Gutman and Richard Nam

Introduction

Goal:

Create a system where a uncategorized reddit post can be suggested to be posted to the most relevant subreddit category

Corrolaries

I. Can features learned from one set of documents be used to infer features on a new set of documents?
II. Is the quality of the classification improved by incorporating information from the proxy measures of post quality/relevance (like the number of upvotes)?

Approach

- Apply different text classification models to the problem
  - Naive Bayes
  - Multinomial Logistic Regression
  - Support Vector Machines
  - Ensemble Methods
- Apply different feature extraction methods to generate input to the models
  - Bag-of-words unigram features
  - Bag-of-words n-gram (2-4) features
  - Weighted average Word2Vec embeddings
  - Doc2Vec (*Paragraph Vector*) embeddings

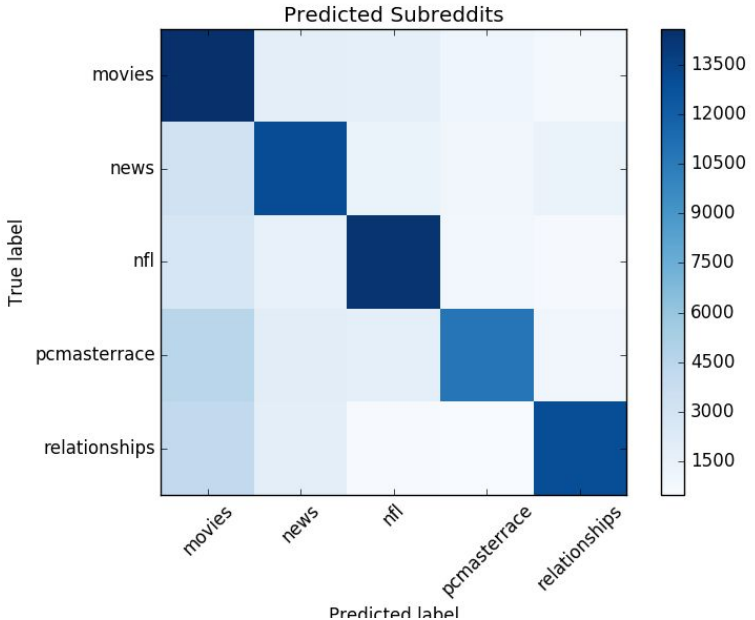
Data

- Reddit post data from Kaggle.com competition
- Originally 1.7 billion comments from May 2015
- Final dataset: 1 million comments drawn 5 popular subreddits
  - NFL, Relationships, News, Movies, PCMasterRace
- Distinct subset of available subreddit categories
- Posts that received more downvotes than upvotes removed
  - Should we train more heavily on higher-scoring posts (weighted bootstrap sample)?
  - Only helps if score is a good proxy for relevance
- Predict data from text only, metadata discarded

Baseline

- 3-gram Naive Bayes
- N-grams with less than 10 counts were dropped
- 494,609 dimensions
- Test accuracy 0.656

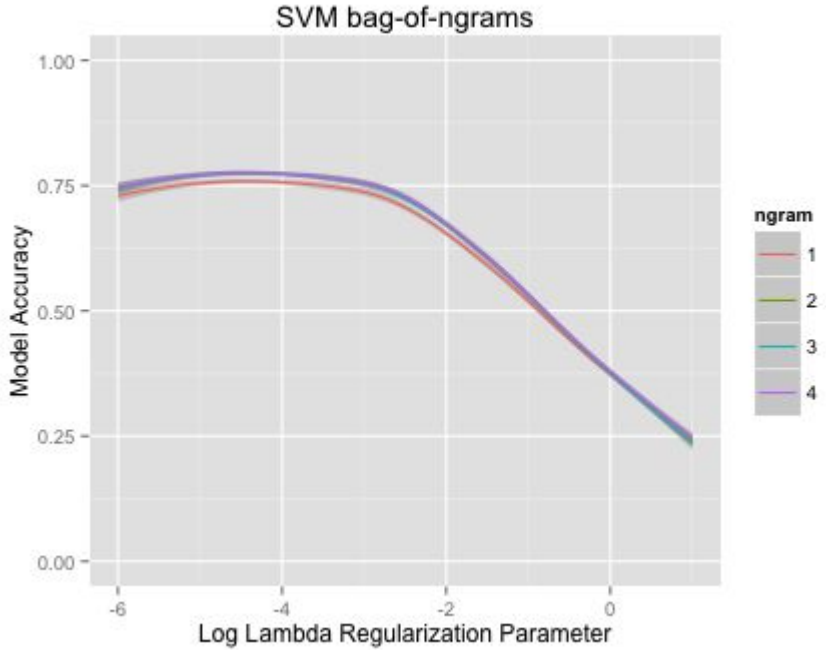
	precision	recall	f1-score
movies	0.50	0.73	0.59
news	0.65	0.65	0.65
nfl	0.72	0.72	0.72
pcmasterrace	0.75	0.54	0.63
relationships	0.78	0.65	0.71
avg / total	0.68	0.66	0.66



Overpredicts movies as subreddit label

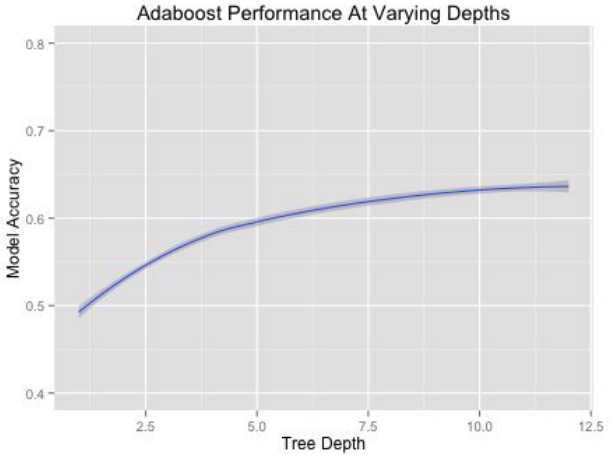
Logistic Regression and Support Vector Machine

- Logistic Regression
  - Multiple n-gram models 1-4
  - Weighted the training set
    - tuned on balance validation set
    - scored on balanced test set
- SVM
  - Multiple n-gram models 1-4
  - Test set score 0.78, 3rd order ngram
  - Tuning regularization hyperparameter C improved accuracy greatly
  - L1, L2, hinge, squared-hinge
    - best: L2, hinge



Ensemble Methods : Adaboost

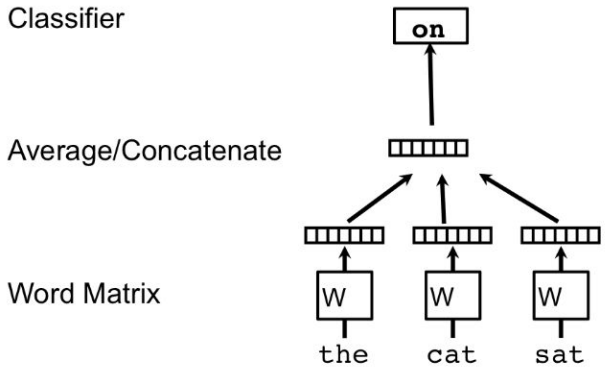
- Try non-linear approach
- Adaboost (incorrect classifications are weighted more heavily in updates)
- Decision Tree classifier
  - increasing depth of trees improves model performance
  - very high-dimensional, sparse text data



- Performance on validation data has not yet leveled off with increasing depth
- Training may require a greater number of estimators
- Explicitly provide weights to predict balanced test set

Documents as Averaged Word Embeddings

- Train Word2Vec model over all words in all documents
  - Remove words that appear less than 10 times across all documents
- Take an (unweighted or weighted) average of all word vectors in each post
  - Weights are from TF-IDF model
  - Stopwords not included in average
- Averaged word embeddings provided as features to the classifier

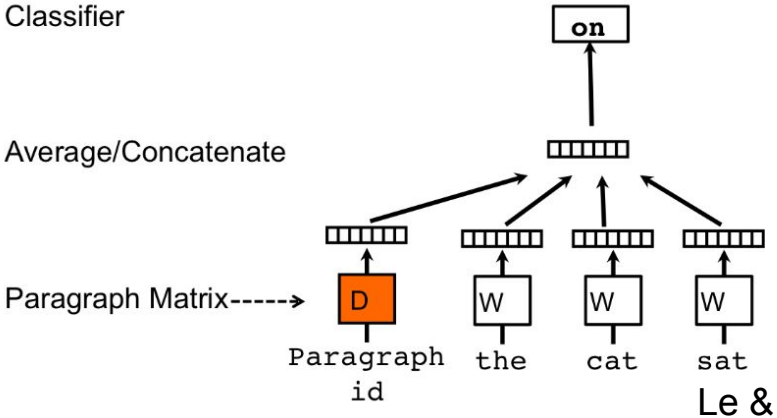


- Upweight vectors corresponding to high tf-idf word embeddings
- Downweight vectors corresponding to low tf-idf word embeddings
- Zero-weight vectors corresponding to English stopwords

Le & Mikolov, 2014

Document Embeddings: Paragraph Vector model

- CBOW Neural network learns two weight matrices simultaneously
  - Word embeddings for every word in the vocabulary, shared across all reddit posts
  - Document embeddings for every Reddit post in the corpus
- Two approaches for learning test set embeddings
  - Train a new Doc2Vec model on the test documents (no labels needed)
  - Use training document embeddings to infer test document embeddings



Follow-up question:
Does concatenating the document and averaged word embeddings provide additional information beyond using either embedding alone?

Le & Mikolov, 2014

Preliminary Results

- N-gram models:
  - **Naive Bayes** *Accuracy:* .65, *Balanced Precision:* .68, *Macro F1: .66*
  - **Logistic Regression** *Accuracy:* .76, *Balanced Precision:* .78, *Macro F1:* .76
  - **SVM** *Accuracy:* .77, *Balanced Precision*: .77, *Macro F1:* .77
  - **Adaboost** *Accuracy* 0.64 (with depth 13 -- still training)
- Embedding models:
  - Use either inferred or separately learned neural networks on test data
  - Preliminary results don't look promising, but parameter tuning needed

Follow-up ideas

  - K-Means Clustering on document embeddings
  - Compute document similiarity within a particular subreddit