Domain Driven Decompositional Semantics

Pranjal Singh
Supervisor: Dr. Amitabha Mukerjee

B.Tech - M.Tech Dual Degree

Thesis Defense

Department of Computer Science & Engineering
IIT Kanpur

June 15, 2015

Outline

Outline

Introduction to Decompositional Semantics

Decompositional Semantics is a way to describe a language entity word/paragraph/document by a constrained representation that identifies the most relevant representation conveying the semantics of the whole.

For example, a document can be broken into aspects such as its tf-idf representation, distributed semantics vector, etc.

Introduction to Decompositional Semantics

Why need Decompositional Semantics?

- It is language independent
- It decomposes language entity into various aspects that are latent in its meaning
- All aspects are important in their own ways

Introduction to Decompositional Semantics

Decompositional Semantics in Sentiment Analysis domain,

- A set of documents $D = \{d_1, \ldots, d_{|D|}\}$
- A set of aspects $A = \{a_1, \ldots, a_{|M|}\}$
- Training data for n ($n < |D|$) documents, $\mathcal{T} = \{l_{d_1}, \ldots, l_{d_n}\}$

Example :

Documents	tf-idf	Word Vector Average	Document Vector	BOW
d_1	0	0	1	0
d_2	0	1	1	0
d_3	1	0	0	1
d_4	×	×	×	×
d_5	1	1	1	1

Using $\mathcal{T}$, D and A, the supervised classifier $\mathcal{C}$ learns a representation to predict sentiments of individual documents.

Problem Statement

Better Language Representation

- To highlight the vitality of Decompositional Semantics in language representation
- To use Distributional Semantics for under resourced languages such as Hindi
- To demonstrate the effect of various parameters on language representation

Contribution of this thesis

Hindi

- Better representation of Hindi text using Distributional semantics
- Achieved state-of-the-art results for sentiment analysis on product and movie review corpus

Paper accepted in regICON'15

New Corpus

- Released a corpus of 700 Hindi movie reviews
- Largest corpus in Hindi in reviews domain

English

- Proposed a more generic representation of English text
- Achieved state-of-the-art results for sentiment analysis on IMDB movie reviews and Amazon electronics reviews

Submitted in EMNLP'15

Contribution of this thesis

Hindi

- Better representation of Hindi text using Distributional semantics
- Achieved state-of-the-art results for sentiment analysis on product and movie review corpus

---

Paper accepted in regICON'15

New Corpus

- Released a corpus of 700 Hindi movie reviews
- Largest corpus in Hindi in reviews domain

English

- Proposed a more generic representation of English text
- Achieved state-of-the-art results for sentiment analysis on IMDB movie reviews and Amazon electronics reviews

---

Submitted in EMNLP'15

Contribution of this thesis

Hindi

- Better representation of Hindi text using Distributional semantics
- Achieved state-of-the-art results for sentiment analysis on product and movie review corpus

Paper accepted in regICON'15

New Corpus

- Released a corpus of 700 Hindi movie reviews
- Largest corpus in Hindi in reviews domain

English

- Proposed a more generic representation of English text
- Achieved state-of-the-art results for sentiment analysis on IMDB movie reviews and Amazon electronics reviews

Submitted in EMNLP'15

Outline

Background on Language Representation

Bag of Words(BOW) Model

- Document d_i represented by $v_{d_i} \in \mathbb{R}^{|V|}$
- Each element in v_{d_i} denotes presence/absence of each word
- **Drawbacks**:
  - High-dimensionality
  - Ignores word ordering
  - Ignores word context
  - Very sparse

Background on Language Representation

Bag of Words(BOW) Model

- Document d_i represented by $v_{d_i} \in \mathbb{R}^{|V|}$
- Each element in v_{d_i} denotes presence/absence of each word
- **Drawbacks**:
  - High-dimensionality
  - Ignores word ordering
  - Ignores word context
  - Very sparse

Background on Language Representation

Term Frequency-Inverse Document Frequency(tf-idf) Model

- Document d_i represented by $v_{d_i} \in \mathbb{R}^{|V|}$
- Each element in v_{d_i} is the product of term frequency and inverse document frequency: $tfidf(t,d) = tf(t,d) \times \log(\frac{\|D\|}{df(t)})$
- Gives weights to terms which are less frequent and hence important
- **Drawbacks**:
  - High-dimensionality
  - Ignores word ordering
  - Ignores word context
  - Very sparse

Background on Language Representation

Term Frequency-Inverse Document Frequency(tf-idf) Model

- Document d_i represented by $v_{d_i} \in \mathbb{R}^{|V|}$
- Each element in v_{d_i} is the product of term frequency and inverse document frequency: $tfidf(t,d) = tf(t,d) \times \log(\frac{\|D\|}{df(t)})$
- Gives weights to terms which are less frequent and hence important
- **Drawbacks**:
  - High-dimensionality
  - Ignores word ordering
  - Ignores word context
  - Very sparse

Background on Language Representation

Distributed Representation of Words(Mikolov et al., 2013b)

- Each word $w_i \in V$ is represented using a vector $v_{w_i} \in \mathbb{R}^k$
- The vocabulary V can be represented by a matrix $V \in \mathbb{R}^{k \times |V|}$
- Vectors (v_{w_i}) should encode the semantics of the words in vocabulary
- **Drawbacks**:
  - Ignores exact word ordering
  - Cannot represent documents as vectors without *composition*

Background on Language Representation

Distributed Representation of Words(Mikolov et al., 2013b)

- Each word $w_i \in V$ is represented using a vector $v_{w_i} \in \mathbb{R}^k$
- The vocabulary V can be represented by a matrix $V \in \mathbb{R}^{k \times |V|}$
- Vectors (v_{w_i}) should encode the semantics of the words in vocabulary
- **Drawbacks**:
  - Ignores exact word ordering
  - Cannot represent documents as vectors without *composition*

Background on Language Representation

Distributed Representation of Documents(Le and Mikolov, 2014)

- Each document $d_i \in D$ is represented using a vector $v_{d_i} \in \mathbb{R}^k$
- The set D can be represented by a matrix $D \in \mathbb{R}^{k \times |D|}$
- Vectors (v_{d_i}) should encode the semantics of the documents
- **Comments**:
  - Can represent documents
  - Ignores contribution of indvidual word while building document vectors

Background on Language Representation

Distributed Representation of Documents(Le and Mikolov, 2014)

- Each document $d_i \in D$ is represented using a vector $v_{d_i} \in \mathbb{R}^k$
- The set D can be represented by a matrix $D \in \mathbb{R}^{k \times |D|}$
- Vectors (v_{d_i}) should encode the semantics of the documents
- **Comments**:
  - Can represent documents
  - Ignores contribution of indvidual word while building document vectors

Background on Sentiment Analysis

- Pang et al.(2004) obtained 87.2% accuracy on a dataset that discarded objective sentences and used text categorization techniques on the subjective sentences
- Socher et al.(2013) used recursive neural network over sentiment treebank for sentiment classification
- Le and Mikolov (2014) use document vector model and obtained 92.6% accuracy on IMDB movie review dataset

Background on Sentiment Analysis

There has been limited work on sentiment analysis in Hindi

- Joshi et al.(2010) used In-language sentiment analysis, Machine Translation and Resource Based Sentiment Analysis to achieve 78.1% accuracy
- Mukherjee et al.(2012) presented the inclusion of discourse markers in a BOW model to improve the sentiment classification accuracy by 2-4%
- Mittal et al.(2013) incorporate hand-coded rules dealing with negation and discourse relations achieving 80.2% accuracy

Background on Sentiment Analysis

There has been limited work on sentiment analysis in Hindi

- Joshi et al.(2010) used In-language sentiment analysis, Machine Translation and Resource Based Sentiment Analysis to achieve 78.1% accuracy
- Mukherjee et al.(2012) presented the inclusion of discourse markers in a BOW model to improve the sentiment classification accuracy by 2-4%
- Mittal et al.(2013) incorporate hand-coded rules dealing with negation and discourse relations achieving 80.2% accuracy

Outline

Outline

Distributed Word Representation

Skipgram

- Each current word acts as an input to a log-linear classifier with continuous projection layer, and predict words within a certain range before and after the current word
- The objective is to maximize the probability of the context given a word:

$$p(c|w;\theta) = \frac{\exp^{v_c \cdot v_w}}{\sum_{c' \in C} \exp^{v_c \cdot v_w}}$$

- v_c and $v_w \in R^d$ are vector representations for context c and word w respectively. C is the set of all available contexts. The parameters θ are v_{c_i}, v_{w_i} for $w \in V$, $c \in C$, $i \in 1, ..., d$

Distributed Word Representation

- Weights between the input layer and the output layer can be represented by a $V \times N$ matrix **W**
- Each row of **W** is the N-dimension vector representation v_w of the associated word of the input layer
- Given a word, assuming $x_k = 1$ and $x_{k'} = 0$ for $k' \neq k$, then
$$h = x^T W = W_{(k,.)} := v_{w_I}$$
$$u_j = {v'}^T_{w_j}.h$$
- v_{w_I} is the vector representation of the input word w_I and u_j is the score of each word in the vocabulary
- There is a different weight matrix **W'**$=\{w'_{ij}\}$ which is a $N \times V$ matrix between hidden and output layer
- Softmax function is used to predict probabilities and Stochastic Gradient Descent is used to update the parameters of the model

Distributed Document Representation

Motivation

- Drawbacks in BOW like sparsity, high-dimensionality, inability to encode context information and consider word ordering
- Composition models alone cannot represent documents (Blacoe and Lapata, 2012)
- Recursive Tensor Neural Networks (Socher et al.,2013) are computationally expensive and cannot be composed into document vectors when there are multiple sentences due to parsing issues
- Presence of similarity measures to deal with synonyms or semantically similar documents

Distributed Document Representation

- Every document is now mapped to a unique vector and id, represented by a matrix D
- Word vector matrix W is shared across all documents and contexts are now separately sampled for each document
- The only difference in this model is that h is now constructed with both W and D.

Semantic Composition

The *Principle of Compositionality* is that meaning of a complex expression is determined by the meaning of its constituents and the rules which guide this combination. It is also known as *Frege's Principle*.
For example,

The movie is funny and the screenplay is good

In the above sentence, consider the word vectors are represented by $w(x)$ and the sentence vector as $S(x)$. Hence,

$$S(x) = c_1 w_1(x) \Theta c_2 w_2(x) \Theta c_3 w_3(x) \Theta c_4 w_4(x) \ldots \Theta c_k w_k(x) \qquad (1)$$

where Θ can be any operation(e.g., addition, multiplication) and c_is are constants.

Semantic Composition

- We describe two approaches to incorporate graded weighting into word vectors for building document vectors.
- Let v_{w_i} be the vector representation of the i^{th} word. Then document vector v_{d_i} for i^{th} document is:

$$v_{d_i} = \begin{cases} 0 & w_k \in stopwords \\ \sum_{w_k \in d_i} v_{w_k} & w_k \notin stopwords \end{cases}$$

  The above equation is 0-1 step-function which ignores contribution of all stop words.
- Another schema which incorporates *idf* weight is:

$$v_{d_i} = \begin{cases} 0 & idf(w_k, d_i) \leq \delta \\ \sum_{w_k \in d_i} idf(w_k, d_i).v_{w_k} & otherwise \end{cases}$$

  where δ is a pre-defined threshold below which the word has no importance and above which the *idf* terms gives importance to that particular word.

Semantic Composition

- We describe two approaches to incorporate graded weighting into word vectors for building document vectors.
- Let v_{w_i} be the vector representation of the i^{th} word. Then document vector v_{d_i} for i^{th} document is:

$$v_{d_i} = \begin{cases} 0 & w_k \in stopwords \\ \sum\limits_{w_k \in d_i} v_{w_k} & w_k \notin stopwords \end{cases}$$

  The above equation is 0-1 step-function which ignores contribution of all stop words.

- Another schema which incorporates *idf* weight is:

$$v_{d_i} = \begin{cases} 0 & idf(w_k, d_i) \leq \delta \\ \sum\limits_{w_k \in d_i} idf(w_k, d_i).v_{w_k} & otherwise \end{cases}$$

  where δ is a pre-defined threshold below which the word has no importance and above which the *idf* terms gives importance to that particular word.

Semantic Composition

- We describe two approaches to incorporate graded weighting into word vectors for building document vectors.
- Let v_{w_i} be the vector representation of the i^{th} word. Then document vector v_{d_i} for i^{th} document is:

$$v_{d_i} = \begin{cases} 0 & w_k \in stopwords \\ \sum_{w_k \in d_i} v_{w_k} & w_k \notin stopwords \end{cases}$$

  The above equation is 0-1 step-function which ignores contribution of all stop words.
- Another schema which incorporates *idf* weight is:

$$v_{d_i} = \begin{cases} 0 & idf(w_k, d_i) \leq \delta \\ \sum_{w_k \in d_i} idf(w_k, d_i).v_{w_k} & otherwise \end{cases}$$

  where δ is a pre-defined threshold below which the word has no importance and above which the *idf* terms gives importance to that particular word.

Semantic Composition

Composition	Accuracy
Multiplication	50.30
Average	88.42
Weighted Average	**89.56**

Table 1 : Results of Vector Composition with different Operations

Method	**Weight**	**Accuracy(1)**	**Accuracy(2)**
0-1 Weighting	0	93.84	93.06
	1	**93.91**	**93.18**
Graded idf Weighting	2	**93.89**	93.17
	2.5	93.87	93.16
	2.8	93.86	93.16
	3	93.86	**93.22**
	4	93.83	93.12

Table 2 : Results on IMDB Movie Reviews(Composite Document Vector);Accuracy(2) is when we exclude tf-idf features

Work Flow

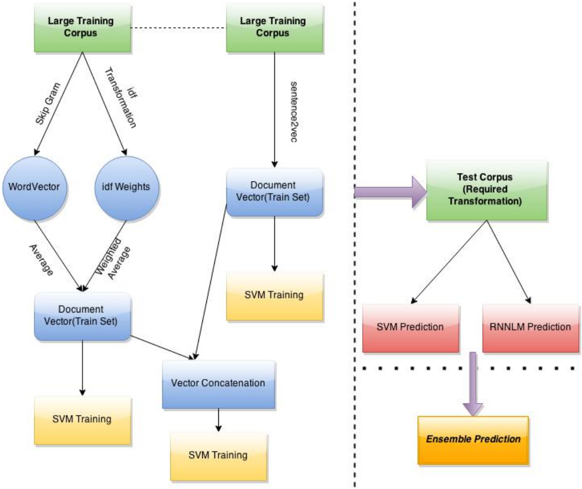

Figure 1 : Work Flow

Outline

Outline

Weightages

Song Features

- Artist similarity has been assigned a weight of 60%
- 20% each for loudness & tempo
- These values have been evaluated to good results

Similarity v/s Popularity

- 65% weightage has been assgined to similarity and 35% to popularity
- A few number of test runs suggested the above weightages to be good

Weightages

Song Features

- Artist similarity has been assigned a weight of 60%
- 20% each for loudness & tempo
- These values have been evaluated to good results

Similarity v/s Popularity

- 65% weightage has been assgined to similarity and 35% to popularity
- A few number of test runs suggested the above weightages to be good

Performance Evaluation

- Most recent t tracks have been considered for testing
- Following m tracks are taken for current mood
- Recommended songs are then matched with the t tracks
- The rank of the top recommendation that appears in the test set is noted
- The similarity of the most similar mood window is also noted

Test Run 1

Similar Users	Mood Length	Weights	Confidence	Rank
50	5	$^1/_3, ^1/_3, ^1/_3$	62.89 %	944
75	5	$^1/_3, ^1/_3, ^1/_3$	48.45 %	3879
100	5	$^1/_3, ^1/_3, ^1/_3$	48.45 %	84
150	5	$^1/_3, ^1/_3, ^1/_3$	51.01 %	135
200	5	$^1/_3, ^1/_3, ^1/_3$	52.63 %	211
50	10	$^1/_3, ^1/_3, ^1/_3$	45.06 %	3418
75	10	$^1/_3, ^1/_3, ^1/_3$	45.50 %	4751
100	10	$^1/_3, ^1/_3, ^1/_3$	46.93 %	1722
50	5	$^1/_5, ^2/_5, ^2/_5$	43.28 %	4033
50	5	$^3/_5, ^1/_5, ^1/_5$	70.57 %	936
75	5	$^3/_5, ^1/_5, ^1/_5$	60.03 %	4367
100	5	$^3/_5, ^1/_5, ^1/_5$	62.10 %	78
150	5	$^3/_5, ^1/_5, ^1/_5$	64.73 %	120

Table 3 : Test Results for Last.FM user: *3en*

Test Run 2

Similar Users	Mood Length	Weights	Confidence	Rank
50	5	$^1/_3, ^1/_3, ^1/_3$	62.39 %	2376
75	5	$^1/_3, ^1/_3, ^1/_3$	50.43 %	N/A
100	5	$^1/_3, ^1/_3, ^1/_3$	50.43 %	7608
150	5	$^1/_3, ^1/_3, ^1/_3$	50.43 %	8828
200	5	$^1/_3, ^1/_3, ^1/_3$	52.40 %	10018
50	10	$^1/_3, ^1/_3, ^1/_3$	48.91 %	N/A
75	10	$^1/_3, ^1/_3, ^1/_3$	48.91 %	N/A
100	10	$^1/_3, ^1/_3, ^1/_3$	48.91 %	N/A
50	5	$^1/_5, ^2/_5, ^2/_5$	43.28 %	N/A
50	5	$^3/_5, ^1/_5, ^1/_5$	70.70 %	2391
75	5	$^3/_5, ^1/_5, ^1/_5$	66.15 %	N/A
100	5	$^3/_5, ^1/_5, ^1/_5$	66.15 %	7095
150	5	$^3/_5, ^1/_5, ^1/_5$	66.15 %	7767

Table 4 : Test Results for Last.FM user: *RJ*

Test Run 3

Similar Users	Mood Length	Weights	Confidence	Rank
50	5	$^1/_3, ^1/_3, ^1/_3$	62.59 %	59
75	5	$^1/_3, ^1/_3, ^1/_3$	50.43 %	607
100	5	$^1/_3, ^1/_3, ^1/_3$	48.37 %	736
150	5	$^1/_3, ^1/_3, ^1/_3$	51.94 %	1095
200	5	$^1/_3, ^1/_3, ^1/_3$	51.94 %	1428
50	10	$^1/_3, ^1/_3, ^1/_3$	48.91 %	2632
75	10	$^1/_3, ^1/_3, ^1/_3$	48.91 %	3736
100	10	$^1/_3, ^1/_3, ^1/_3$	46.91 %	4304
50	5	$^1/_5, ^2/_5, ^2/_5$	43.28 %	563
50	5	$^3/_5, ^1/_5, ^1/_5$	70.94 %	85
75	5	$^3/_5, ^1/_5, ^1/_5$	66.15 %	555
100	5	$^3/_5, ^1/_5, ^1/_5$	63.88 %	650
150	5	$^3/_5, ^1/_5, ^1/_5$	64.59 %	970

Table 5 : Test Results for Last.FM user: *eartle*

Test Run 4

Similar Users	Mood Length	Weights	Confidence	Rank
50	5	$^1/_3, ^1/_3, ^1/_3$	61.84 %	141
75	5	$^1/_3, ^1/_3, ^1/_3$	49.83 %	629
100	5	$^1/_3, ^1/_3, ^1/_3$	49.71 %	674
150	5	$^1/_3, ^1/_3, ^1/_3$	51.10 %	4351
200	5	$^1/_3, ^1/_3, ^1/_3$	51.48 %	4363
50	10	$^1/_3, ^1/_3, ^1/_3$	47.49 %	3160
75	10	$^1/_3, ^1/_3, ^1/_3$	47.49 %	3135
100	10	$^1/_3, ^1/_3, ^1/_3$	47.54 %	3225
50	5	$^1/_5, ^2/_5, ^2/_5$	43.28 %	4422
50	5	$^3/_5, ^1/_5, ^1/_5$	67.74 %	103
75	5	$^3/_5, ^1/_5, ^1/_5$	66.18 %	470
100	5	$^3/_5, ^1/_5, ^1/_5$	64.43 %	471
150	5	$^3/_5, ^1/_5, ^1/_5$	64.43 %	5227

Table 6 : Test Results for Last.FM user: *franhale*

Test Run 5

Similar Users	Mood Length	Weights	Confidence	Rank
50	5	$^1/_3, ^1/_3, ^1/_3$	62.64 %	4953
75	5	$^1/_3, ^1/_3, ^1/_3$	50.09 %	9857
100	5	$^1/_3, ^1/_3, ^1/_3$	50.09 %	11647
150	5	$^1/_3, ^1/_3, ^1/_3$	51.10 %	6587
200	5	$^1/_3, ^1/_3, ^1/_3$	51.48 %	8008
50	10	$^1/_3, ^1/_3, ^1/_3$	48.08 %	N/A
75	10	$^1/_3, ^1/_3, ^1/_3$	48.08 %	2584
100	10	$^1/_3, ^1/_3, ^1/_3$	48.08 %	2887
50	5	$^1/_5, ^2/_5, ^2/_5$	43.28 %	N/A
50	5	$^3/_5, ^1/_5, ^1/_5$	70.75 %	5005
75	5	$^3/_5, ^1/_5, ^1/_5$	65.97 %	7819
100	5	$^3/_5, ^1/_5, ^1/_5$	65.97 %	9345
150	5	$^3/_5, ^1/_5, ^1/_5$	68.19 %	5749

Table 7 : Test Results for Last.FM user: *massdosage*

Optimizations

- Parallelization: Independent jobs have been forked in parallel to reduce runtime
- On-Demand Caching: Not only avoids loading the entire DB into memory, but also prevents disk access each time the same resource is called for. Also reduces multiple file accesses
- Minimal data handling: Minimal data is stored in memory in a serialized JSON format

Future Work

- Larger and newer dataset
- Machine learning to implement feedback mechanism for user specific weightages
- More features like MFCC can be included appropriately
- Code can be optimized even further by the use of distributed systems

Thank you!