Product Knowledge Graph from User Reviews and Recommendations

Jatin Arora
13CS10057

09 November 2016

A report submitted for
B.Tech. Project Part 1
in
Computer Science and Engineering

Guides:
Prof. Pawan Goyal
Dr. Sayan Pathak

ACKNOWLEDGEMENT

I am deeply grateful to my supervisor Prof. Pawan Goyal and Dr. Sayan Pathak for giving me the opportunity to work on this project. I am thankful to their aspiring guidance, invaluably constructive friendly advice during the course of this project. I am sincerely grateful to him them sharing their truthful and illuminating views on a number of issues related to the project.

I am also thankful to my teachers of the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, who have instilled in me the scientific spirit of inquiry, experimentation, observation and inference, without which I would not have been able to produce this work. I hope I will be able to rise up to the high standards set by my predecessors in this department.

-Jatin Arora

CERTIFICATE

This is to certify that the report entitled "Product Knowledge Graph from User Reviews and Recommendations" submitted by Jatin Arora, in the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India, for the award of the degree of Bachelor of Technology, is a record of an original research work carried out by him under my supervision and guidance. This report fulfills all the requirements as per the rules and regulations of this institute. Neither this report nor any part of it has been submitted for any degree or academic award elsewhere to the best of my knowledge.

Prof. Pawan Goyal
Dept. of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Kharagpur, India, 721302

Contents

1 Introduction

People often look for products with some desired features. While selecting a product, many things are taken into account, such as, what features, how does the product perform under certain conditions when compared to other products etc. During selection, people compare the products on some selected attributes of their choice. Very often values of these attributes are mentioned in the product specifications itself such as, for a camera, the price, weight ,megapixels etc. are clearly written. But sometimes users are also interested in a number of attributes which are subjective in nature and cannot be measured as such. For e.g. auto-focus speed of a camera, noise performance, low light performance etc. Thus a user simply cannot make comparison between two products on these features based on the specifications only. In such cases the user is bound to take opinions from some other users who have used both the products and have enough knowledge about them.

1.1 Brief description of a Product Review

Various e-commerce websites allow users to post their experience after using a product. In these reviews, the users usually express their opinions about the product, the features they liked and if any feature needs improvement. Users also compare among different products and make alternate product recommendations to the readers. Usually a rating is also associated with these reviews where the users score the product out of 5 or 10 points. The websites also allows the readers to vote on these reviews. So a reader up-votes a review if he finds it helpful. So the ratio of total up-votes and total votes is said to be the **helpfulness score** of a review.

1.2 Some observations on helpfulness score

A study on 50,000 reviews which had at least one vote
Avg. helpfulness score for all reviews = **0.67**
Avg. helpfulness score for reviews listing pro-cons = **0.81**
Avg. helpfulness score of reviews with at least one other product = **0.78**

Thus we can claim that reviews describing the various features of a product or mentioning it's pro-cons or making a comparison with other products are more helpful to the readers.

2 Problem Description

Based on product reviews given by users on various e-commerce websites and blogs, we want to extract various information about a product such as what are it's important features, how good are theses features and how useful do users find them. We also aim to extract the relationships among various products based on their features, e.g. two products have similar price but, one is better than other in terms of a particular feature. Thus, the idea is to construct a product knowledge graph where the nodes are the products and edges are a relationship between them. This graph can be thought of as a collection of (e1, r, e2) pairs, where e1 and e2 are two entities and r is a semantic relation among a fixed set of possible relations.

Example: For a knowledge graph of cameras, nodes can be say, (Nikon Coolpix S8100) and (Cannon PowerShot), and say, Nikon has better red eye reduction than Cannon as per the reviews, so, we have an edge directed from Cannon to Nikon with a weight signifying how much better Nikon is than Cannon in terms of red eye reduction.

3 Procedure Overview

A brief overview of the steps in making the product knowledge graph

1. **Data collection:** Crawling product reviews from e-commerce websites as well as blogs. We also extract some useful meta-data associated with reviews as well such as the rating, up-votes, total votes.

2. **Sentiment analysis of the user reviews** : Identify review sentences where a product comparison is done. Then in any such sentence, identify the products under comparison, the aspect/feature of the product being compared and users' opinion about that feature. Finally, also capture the direction of opinion among the products in the review.

3. **Preparation of the knowledge graph** : From the information extracted above, for each product make a node in the graph. For every comparison based review sentence identified make an edge between the products under comparison depicting the feature being compared. Device a mapping from opinion words to weights of theses edges in the graph.

4. **Query the graph:** With the product knowledge graph at our disposal, providing a algorithm to order the search results according to query relevance.

4 Dataset Used

Websites from which data was crawled : Amazon.com and Amazon.in
Domain : Cameras and Electronic Gadgets
Amazon.com
Total no. of products for which reviews crawled = 1260
Total no. reviews = 1,26,410
Total no. of users proving reviews = 70,023
Amazon.in
Total no of products for which reviews crawled = 50
Total no reviews = 5,532
Total no of users proving reviews = 4,329

Another completely annotated data set was used (Jindal and Liu, 2006b)[1] to test our approaches. It had around 500 reviews on various products such as cameras, mobile phones, PC, and ipads.The reviews were labeled with feature/aspect, opinion and compared entities.

Example Review:
product/title: Canon EOS 5D 12.8 MP Digital SLR Camera (Body Only)
review/score: 4.0
review/text: This camera is superior in almost every way. I quibble a bit about the location of some of the controls. For example, the on/off switch is not in a great location, Nikon has a better location for this on its cameras. This type of complaint aside, which is entirely subjective in nature, You can't find anything better.
review/helpfulness: 3/7
review/profileName: Frank A. Langheinrich
product/productId: B0007Y791C
review/summary: Canon 5D
review/time: 1178928000
review/userId: A1BANMTXGPYT7M

5 Baseline Approach

How to extract a relationship from a review

- **Pre-processing** Split a preview into sentences. Stem and tokenize the words. Perform POS tagging for the words.
- **Identify features / attributes** of the products discussed in the reviews. Reviews are often compared on some features. Sometimes the user may not explicitly mention the attributes while comparing two products. For e.g. P1 cheaper than P2. Here the comparison is based on price, but not explicitly mentioned.
- **Identify a product** mentioned in a review. User may mention the whole name of product or just the model number. Also user may use pronouns like "this" or "it" to reference the product for which he is writing the review.
- **Identify the opinion** user mentioned for that particular aspect.
- **Extract relationships** from the reviews. Identify instances where the user mentions products, aspect and a opinion .
  E.g. Sentence : " Canon D1 is has better image quality than Nikon 500D ".

  Entity+ = Canon D1
  Entity- = Nikon 500D
  Aspect = image quality
  Opinion = better
  Relationship = (Canon D1, image quality , better , Nikon 500D)

5.1 Identifying features/aspects in a review

Going through sample reviews , blogs , camera specific sites , a list of popular attributes for camera domain was created. These features usually associated with a specific functionality of a camera and are most likely to mentioned in a review wherever there is a comparison among products. Eg. megapixel, expensive, compact, storage space, low light sensitivity, buffer, zoom, noise reduction, display etc.

A total of 55 attributes were listed. We present two ways identify an aspect, one with the dictionary matching where we match aspects from our aspect list in a review sentence. We also present a machine learning approach where we consider all Noun and verb words in a sentence as possible aspects and use a binary classifier to classify them as aspect and non aspect.

5.2 Identifying a product in a review

An exhaustive list of camera models was made from e-commerce websites. Users may not always write the complete name of a product, instead they may write just the model number (an alpha-numeric string) in a review to mention a product. So a mapping for each product to its unique product code which a user is most likely to mention was created.
E.g. Canon Power Shot A2300 – A2300.
Sometimes, product name is not explicitly mentioned, but just a reference is made, like, 'This camera is better...'. We identify two product entities, one is the entity+ and entity-. Entity+ is the product that usually comes first in the comparison, although not necessarily. For eg. D500 is better than 600D. Here entity+ is D500 and entity- is 600D.

5.3 Identifying opinion

Once the aspect and comparing entities are identified, depending on the position of the aspect and entities, we extract the opinion words by matching a part of a sentence to a POS pattern.

5.4 Filtering Reviews

We only consider reviews where user mentions at least one other product.
Assumption : User is making a comparison between the product for which review is being written (model1) and the other product. (model2)
Filtering:
Total Reviews in database : around 1.2 lakh
Total Reviews where other product name and adjective = 1657
Total potential sentences with other product name = 8696
Total potential sentences with other product name and adjective = 3550

Out of 1657 Reviews:

training reviews = 1000
test reviews = 657

Training Set:
potential statements with other product name and adjective : 2155

Test set
Potential test cases from Amazon.com : 657
Potential test cases from amazon.in : 80

5.5 Extracting Relationships using Hand-Built Patterns

Relationship = (entity+ , aspect , opinion , entity-)

- Identify some common patterns which may fit a sentence where a comparison is being made with other product. There could be a number of possibilities like both the aspect and opinion comes before models in the sentence or one comes before and the other comes after . These patterns should be able to capture a relationship with fair accuracy at the same time they should be generalized and should not over fit.
  E.g. model1 (Adjective) (Aspect) model2.
  Canon D9 has better lens than Nikon D1.

- Once a Review Sentence fits a pattern, set entity+ and entity- according to the pattern .

- Identify the aspect on which comparison is being made from appropriate section of the sentence as specified by the pattern.

- Identify the opinion word that describes the aspect in the relationship. It is usually an adjective such as better , more. This is done by matching a part of sentence with a POS pattern.

5.6 Patterns Used

model1 = for which review was written. 'This' or 'it' is a reference to model1
model2 = another model that occurred in the review

5.6.1 POS Patterns for Opinion finding

- **Order :** (ADJ | VERB | ADVERB | DET | PREP)*
  with at least one ADJ or ADVERB
  Eg. "Very high Quality" is a match to this pattern

- **Order :** (VERB) (ADJ | ADVERB | DET | PREP)*
  with at least one ADJ or ADVERB
  Eg. "is way better", "has higher".

5.6.2 General Patterns

- **Order :** (Aspect and Opinion section) (Preposition) model2
  Body : (String1) (than | as | to | over | compared | from | of) (String2) model2
  e.g. **this** camera has **far better image quality** than **D5**.
  e.g. **Image quality is bar better** in **this camera** than **D5**.

- **Order** model2 (VERB) (Aspect Opinion subsection) *optional (Preposition) model1
  Body model2 + (String1)
  e.g. **D5** has **sharper zoom** (**than D1.**)
  e.g. In **D5** the **zoom is much sharper** (**than D1.**)

- **Order:** (Aspect) (Preposition) (model) (Opinion section)
  Body : (String1) (of | in) (model) (String3)
  e.g. The **zoom** in **D9** is **far better**

- **Order:** (Opinion Section) (Preposition) model (Aspect)
  Body :(String1) (than | as | to | over | compared | from | of) model (String 2)
  e.g. **D1** is **very similar** to **D9** in terms of **low light sensitivity**.

6 Results:

6.1 In Training set:

Total number of potential sentences with a relationship in training set: **2155**
Total captured by the patterns : **697**

Pattern wise Performance:

Pattern Number	Total Relationships Captured
1	345
2	251
3	41
4	60

6.2 In Test Set:

Using Amazon.com data as test set:
Total number of reviews : **1000**
Total number of potential sentences with a relationship in training set: **1415**
Total captured by the patterns : **355**

Rough Precision Estimate:
Out of 100 relationships random chosen tagged by automated Tagger:
Total cases where aspect-opinion pair correct : **85**
Total cases with aspect-opinion pair and Entity+ , Entity- correct : **68**
Precision - 68%

Rough recall Estimate:
Out of 100 manually tagged relationships:
Cases correctly captured by automated tagger : 52
Recall - 52%

6.3 How many times a particular camera model was compared

For data in training set, number of times the models occurred in comparisons. In our product knowledge graph , models are the nodes and comparisons are the directed edges between them.
Node Degree Distribution: (Total nodes = 1160)

Degree Range	Total number of nodes
greater than 30	8
20-30	8
10-20	13
5-10	20
1-5	97
0	1017

This shows that some camera models are very popular and are compared more often then others. The models in high degree range have more probability of being compared with another camera.

6.4 Which were the most compared aspects:

Edge Label frequency :
version :142
weight :65
price :62
focus :40
sensor :25
lens resolution :21
size :21

7 Machine Learning based approach to identify aspects and entities

In the pattern based approach, we had a list of aspects and product code numbers. There we matched values from lists to words in sentences to identify aspects and models. So it was a dictionary matching based approach. The limitations of this approach is that the dictionary has to provided. So here we present another approach which don't require pre-defined dictionaries.

7.1 Identifying an aspect

Our main idea here is to consider each Noun and verb word in a sentence as a potential aspect. Inspired from the work of camera comparison slr by Kessler

and Kuhn [2], we train a binary classifier with some context and dependency parse features. The list of features considered are:

- 2 word window around the given word as context
- POS of word-1, word-2, word+1, word+2
- Relationship with parent in dependency parse tree
- POS of parent in dependency parse tree
- Relationship with children in dependency parse tree
- POS of children in dependency parse tree

7.2 Results for Aspect classification

We tested the aspect classification task on two set of data set.The first data set was our own Amazon.com reviews which was labeled by the hand built patterns. The other data set is the Jindal and Liu [1] data set which is completely labeled.We compare our results on Jindal and Liu data set with the result obtained by Kessler and Kunh[2] on the same data set.

The overall procedure to prepare the test set is:

1. For each review sentence in the data set, split in into words and tokenize it.Use Stanford NLP POS tagger to obtain POS tags and Dependency Parser to obtain dependency parse tree.
2. Consider each word as a potential aspect. The word should be either noun or verb and must be at least 4 characters long.
3. Obtain the mentioned features for the word.
4. If the word was labeled as aspect, set it as a Yes instance in the test data set. Else set it as a No instance.
5. Repeat the above steps for word bi-grams as well. The only change is that for the word bi-gram to be considered as an aspect, the POS of both words should be Noun, Noun or Adj, Noun.

6. Obtain the features for the bi-gram. For dependency parse features consider the first word as the candidate word for parent, children relationships.

7. If the bi gram was labeled as aspect, set it as 'Yes' instance. Else if the bi-gram itself is not an aspect and both the words individually were not labeled as aspect, set it as 'No' instance.

On the test data set, we perform 5 fold cross validation using various ML algorithms such as Naive byes, decision trees, SMO.

Next We present the results for Amazon data.The total number of yes instances were 462 and No instances were 1842. We focus on the results for Yes instances.

Algorithm	Precision	Recall	F-Measure
Naive Bayes	0.439	0.344	0.385
SMO	0.604	**0.47**	**0.52**
Decision Tress	**0.687**	0.171	0.27
Bayesian nets	0.488	0.264	0.342

Results on Jindal and Liu data set compared with SRL method by Kessler and Kunh. Total number of 'Yes' instances were 398 and 'No' instances were 4302. We present the results after 5-fold cross validation using various algorithms.

Algorithm	Precision	Recall	F-Measure
SRL	0.58	0.30	0.40
Naive Bayes	0.359	0.337	0.347
SMO	0.599	**0.402**	**0.48**
Decision Tress	**0.648**	0.176	0.27

Results from SMO out performs the SRL method.

7.3 Discussion on the results

There are many ways to express a comparison and the size of the available training data is relatively small. Since reviews can be very unstructured, the

same thing can be written in various ways in reviews such as:
1.**Image Quality** is far better in this camera.
2. In this camera, the only improvement is **image quality**.

Thus there is no specific way to identify an aspect and thus we use a combination of features. Usually the near by opinion words help us to identify an aspect. We encounter low recall values since not all aspects follow a particular set of features. Most critical are those cases where the opinion word is very far away from aspect and hence doesn't help much to identify the aspect. For e.g. consider this sentence:
Of all things that is better in this camera , what matters most is **FPS rate.**
Such statements present us a challenge to improve the recall rate.

8 Structural Alignment for Data-set Expansion

We observe that in order to get a better classification accuracy using the standard classification techniques applied above or for applying deep learning, we require a large amount of labeled data. It is not possible to manually annotate such a large amount of user reviews. But since the higher-level semantic structure of comparisons as they appear in reviews is clear-cut, it could respond favorably to weakly supervised training strategies that start out from a seed set of manually annotated data.(Kessler and Kuhn, 2015)[3] Structural Alignment has been seen to be useful in Semantic Role-based Labeling[5] for data expansion and since, comparisons can be mapped to predicate-argument structure, identification of comparisons can be seen as a SRL problem.

8.1 Approach

The hypothesis is that predicates that appear in a similar syntactic and semantic context will behave similarly with respect to their arguments so that the labels from the seed sentences can be projected to the unlabeled sentences. These newly labeled sentences can then be used as additional training data.

8.2 Procedure Outline

Given a small set of labeled seed sentences (seed corpus) and a large set of unlabeled sentences (expansion corpus), we expand on the predicates identified in the seed sentences. For a predicate 'p' of a seed sentence 's', for every unlabeled sentence 'u',

- **Sentence Selection:** Consider 'u' if and only if it contains a predicate compatible with 'p'.
- **Argument Candidate Creation:** Get all argument candidates from 's' and from 'u'.
- **Alignment Scoring:** Score every possible alignment between two argument candidate sets.
- Store best scoring alignment and its score if and only if at least one role bearing node is covered.
- Finally for every predicate, take up the top 'k' sentences labeled by each predicate 'p' and project the labels from the arguments in the seed sentence 's' to get newly labeled sentences.

8.3 Creation of Argument Candidates

Kessler and Kuhn, 2015 [3] have proposed two methods to select the argument candidates:

- **Dependency-Filtered:** Use all ancestors of the predicate until the root and their direct descendants plus all descendants of the predicate itself. Remove all conjunctions and prepositions and add their direct children to the candidate set.
- **Path-Filtered:** Take paths from the predicate to each argument in the labeled sentence and search for the exact same paths in unlabeled sentence. All nodes on the path are extracted as candidates.

8.4 Similarity Measures and Alignment Scoring

Similarity of an alignment between two sentences 's' and 'u' is the averaged sum of all word alignment similarities, themselves the averaged sum of different word alignment similarity measures.

Similarity Measures:

- **Sim-vs:** Cosine similarity of co-occurrence vectors of 2 words
- **Sim-neigh:** (Sim-vs of left neighbor + Sim-vs of right neighbor) / 2
- **Sim-dep:** Dependency relation similarity + POS Similarity
- **Sim-tok:** Similarity of distance (No. of tokens) of candidate from predicate
- **Sim-lev:** Similarity in number of 'ups' and 'downs' in dependency tree path from argument to predicate
- **Sim-path:** Average Sim-dep of all words on path from argument to predicate

Based on these, two combinations of similarity measures are prepared:

- **Flat Similarities:** Sim-vs, Sim-dep
- **Context Similarities:** Sim-vs, Sim-dep, Sim-neigh, Sim-tok, Sim-lev, Sim-path

8.5 Data Expansion Approaches

Based on the method chosen for candidate creation and variation of similarity measure used for scoring the candidate alignments, Kessler and Kuhn, 2015 propose four versions of expansion:

- **Path-Flat:** Path-Filtered candidate creation and Flat similarities
- **Path-Context:** Path-Filtered candidate creation and Context similarities
- **Dep-Flat:** Dependency-Filtered candidate creation and Flat similarities
- **Dep-Context:** Dependency-Filtered candidate creation and Context similarities

It is observed that **Dep-Context** gives the best expansion results with the performance being slightly improved over the non-expanded original sample. Hence, for our data-set we use this approach for corpus expansion.

8.6 Preparation of Word Embeddings

Image and Audio processing systems work with rich high-dimensional datasets encoded as vectors of individual raw pixel-intensities for image data, power spectral density coefficients for audio data. So, all the required information is available in the dataset itself. But, this is not so for NLP Tasks as here, words are treated as discrete atomic units and so simply the word itself cannot capture relationships among them. So, we go for a richer representation for words as k-dimensional vectors. Two standard approaches for this are:

- **Count-based approachs**, which compute stats of how often some word co-occurs with its neighbor words and then reduce the dimensions of the co-occurrence matrix to desired k-dimensional representation for each word.
- **Predictive Models** directly try to predict a word from its neighbors and encode this information as small dense embedding vectors.

For structural alignment and again later for applying any deep learning approach, we prepare word-embeddings for **Reviews Corpus on Electronic Gadgets** from **Amazon**.

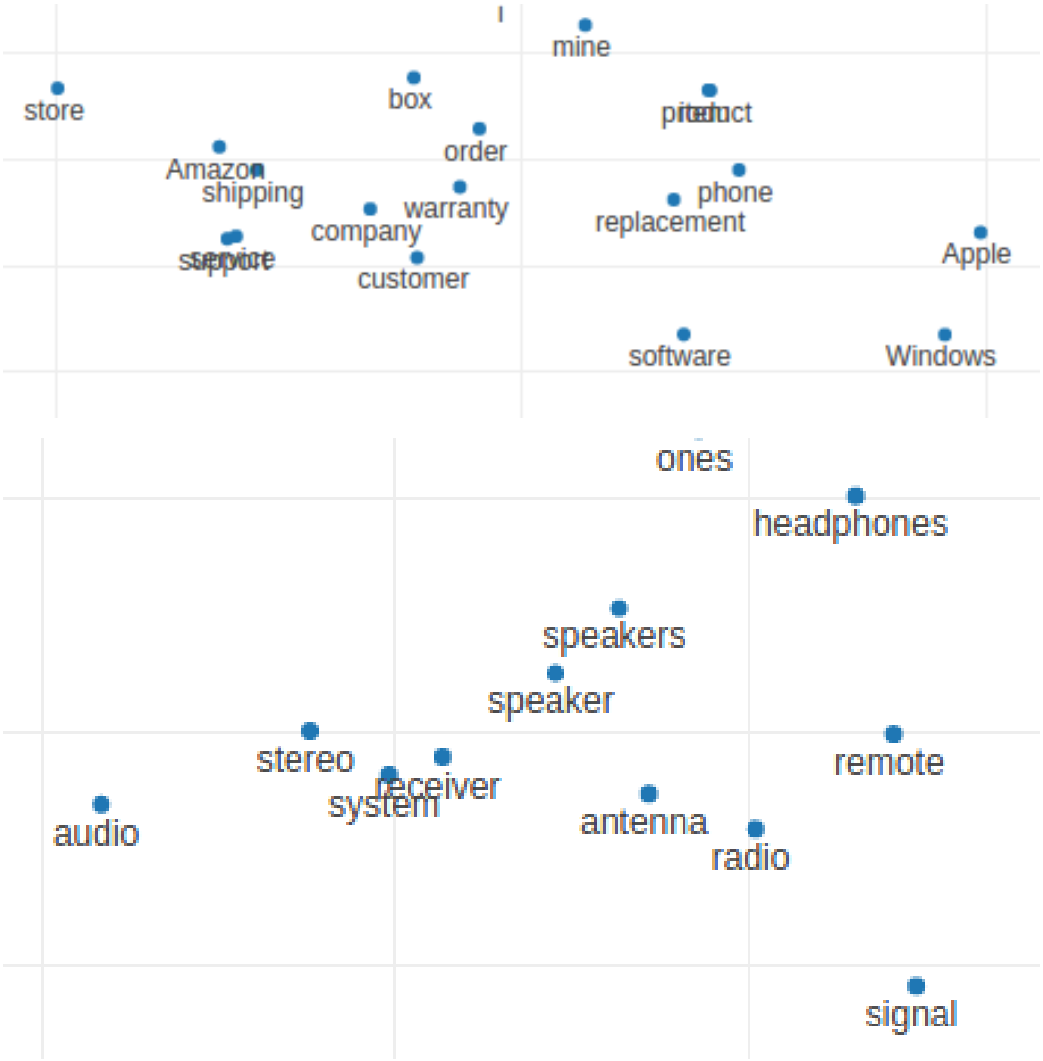
Approaches:

1. Word2Vec using Tensorflow
2. GloVe
3. Context Vectors creation by University of Stuttgart

We used these approaches with varying vector sizes, 100, 1000, 2000.

8.7 t-SNE Projections for Word Embeddings

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets. We have used this to visualize our word embeddings developed using different approaches. Example: Two magnifications of t-SNE Projections for Vector-Size = 50 using Word2Vec.



8.8 Setup

- We manually labeled **500** comparison based sentences from Electronics Corpus with Entity+, Entity-, Opinion, Predicate tags. Also we obtained manually tagged sentences on Camera Corpus by Kessler and Kuhn, 2014 [9]
- We filtered **30000** sentences from remaining reviews in Electronics Corpus which had at least one comparative adjective or adverb i.e. 'JJR' or 'RBR' POS Tag.

- We passed both, the labeled and unlabeled sentences to **MATE Dependency Parser**[6][7] and for the labeled sentences added additional fields for the labeled information to the dependency parse obtained.
- Hence prepared labeled and unlabeled sentences are then fed to the system for automated labeling.
- To make it efficient, we did the **processing in paralled** by dividing the unlabeled corpus into **chunks** of approx. **4000** sentences each and ran it on departmental servers (both cpu's and gpu's) having 24 or 36 processors each.

8.9 Evaluation Procedure and Setup

- To test how good the automated labeling is, we ran the best reported algorithm which captures word dependencies with context on different word embeddings prepared.
- Input labeled dataset size fixed: 360 sentences
  Unlabeled dataset size: 3000 sentences with at least one 'JJR' or 'RBR' POS Tag
- We randomly picked **400** sentences and manually checked whether entity, aspect and predicate labeling is correct.
- If aspect is correct, we assign a '+1' for the sentence, otherwise a '0'(zero). Similarly for the entities and predicates as well.

8.10 Results

	Entity-1		Aspect		Predicate		Entity-2	
K	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
5	0.65	0.74	0.70	0.72	1.00	1.00	0.45	0.72
10	0.59	0.68	0.68	0.72	1.00	1.00	0.45	0.58
15	0.58	0.69	0.67	0.71	1.00	1.00	0.43	0.60
20	0.53	0.63	0.64	0.68	1.00	1.00	0.43	0.61

Table 1: Precision and Recall for Default Word Embeddings with Context Window Size = 2 and Vector Size = 2000

	Entity-1		Aspect		Predicate		Entity-2	
K	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
5	0.65	0.74	0.67	0.71	0.95	1.00	0.40	0.63
10	0.59	0.68	0.65	0.69	0.98	1.00	0.46	0.63
15	0.53	0.63	0.61	0.67	0.98	1.00	0.42	0.67
20	0.53	0.61	0.60	0.68	0.99	1.00	0.41	0.69

Table 2: Precision and Recall for Electronics Reviews Word Embeddings with Context Window Size = 3 and Vector Size = 2000

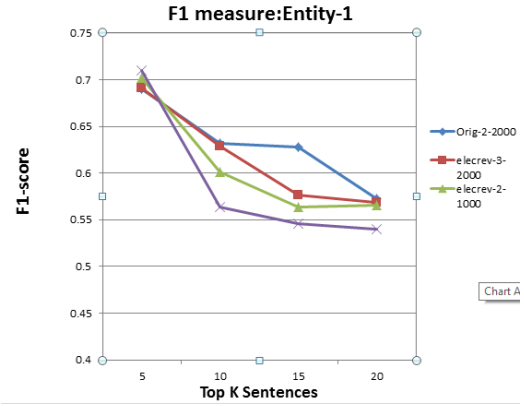
	Entity-1		Aspect		Predicate		Entity-2	
K	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
5	0.68	0.73	0.67	0.71	0.98	1.00	0.43	0.63
10	0.56	0.65	0.63	0.67	0.99	1.00	0.41	0.58
15	0.53	0.61	0.61	0.63	0.99	1.00	0.44	0.64
20	0.53	0.61	0.62	0.67	0.99	0.99	0.43	0.65

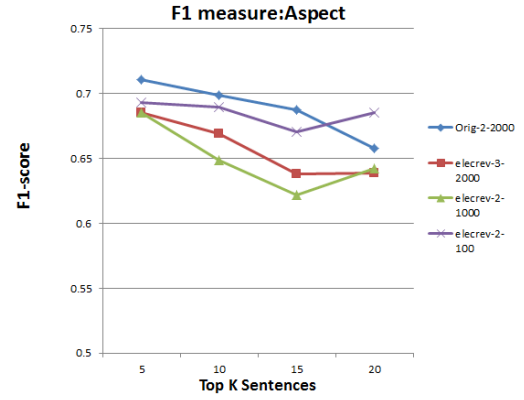
Table 3: Precision and Recall for Electronics Reviews Word Embeddings with Context Window Size = 2 and Vector Size = 1000

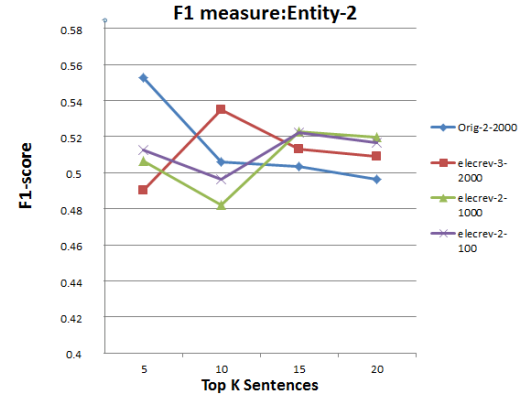
	Entity-1		Aspect		Predicate		Entity-2	
K	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
5	0.65	0.78	0.67	0.72	0.95	1.00	0.45	0.60
10	0.51	0.63	0.65	0.73	0.98	1.00	0.44	0.57
15	0.49	0.61	0.62	0.73	0.98	1.00	0.44	0.64
20	0.49	0.60	0.63	0.75	0.99	1.00	0.43	0.65

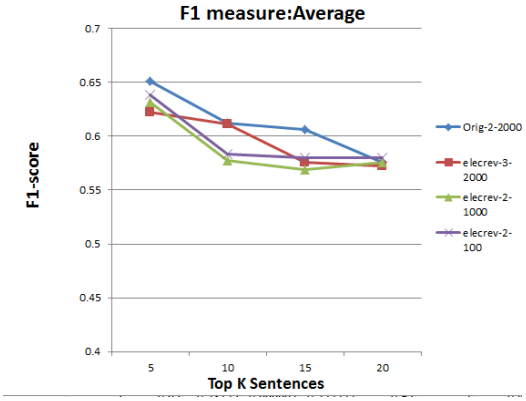
Table 4: Precision and Recall for Electronics Reviews Word Embeddings with Context Window Size = 2 and Vector Size = 100

We calculated the F1-Score separately for Entity and Aspect Detection and then a combined average also, and plotted the following curves:


F1 measure:Entity-1
F1-score
Top K Sentences
Orig-2-2000
elecrev-3-2000
elecrev-2-1000
Chart A


F1 measure:Aspect
F1-score
Top K Sentences
Orig-2-2000
elecrev-3-2000
elecrev-2-1000
elecrev-2-100

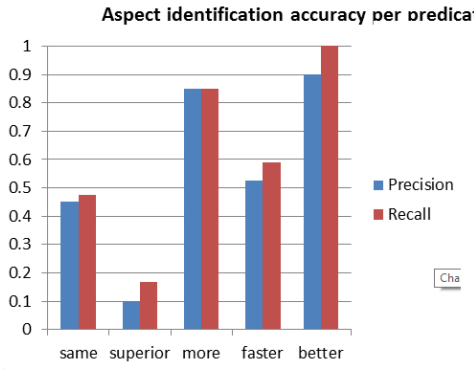

F1 measure:Entity-2
F1-score
Top K Sentences
Orig-2-2000
elecrev-3-2000
elecrev-2-1000
elecrev-2-100



8.11 Observations and System Limitations

- **Aspect Identification:** With embeddings of size 100, prepared from electronics corpus considering context window of size 2, we get a F1-score of approx. 69% for 20-fold expansion.
- **Entity-1 Identification:** With embeddings of size 100, prepared from electronics corpus considering context window of size 2, we get a F1-score of approx. 71% for 5-fold expansion, while for 20-fold expansion all approaches give almost same F1-score of approx. 56%.
- **Entity-2 Identification:** With embeddings of size 100, prepared from electronics corpus considering context window of size 2, we get a F1-score of approx. 52% for 20-fold expansion.
- **Overall:** If we give equal weight-age to entity and aspect identification, in best case, we achieve F1-Score of 65% with original embeddings of size 2000 considering context window of size 2, while for 20-fold expansion, all approaches report amost same overall F1-score of 57%. Electronics Review Embeddings of size 100 produce good recall in general.
- Entity-2 Identification precision is lesser as compared to others. This is because many times, the labeled seed sentences have an Entity-2 but many user reviews do not explicitly mention another product name but the system always marks some word as Entity-2 since it was there in the seed sentence.

- Aspect and Entity Identification varies with predicates, as seen in the graph below for Aspect Identification. This is because of different contexts in which these predicates are used.



- There are sentences like, 'Nikon has better image quality than Cannon and Sony'. Here, predicate 'better' is used for multiple Entity2's, i.e. 'Cannon' and 'Sony'. Such dependencies are not captured by the system.

- For aspects which are not uni-grams like 'low light resolution', the system takes in only the root word i.e. 'resolution', as it considers only unigrams as aspects.

9 Deep Learning Approach

Our task of identifying product entities and aspects can be thought of as a sequence labeling task, where the sequence is the words of the sentence and we have to label some words with the labels 'ENTITY1', 'ENTITY2', 'ASPECT' and 'OPINION', while all the rest of the words will be marked 'NONE'.

Example:
Sequence1: Nikon Coolpix has better image quality than Cannon PowerShot.
Sequence2: ENTITY1 ENTITY1 NONE OPINION ASPECT ASPECT NONE ENTITY2 ENTITY2.

This type of sequence to sequence mapping can be handled well by Long Short Term Memory(LSTM) based Recursive Neural Networks(RNN).

9.1 Using LSTMs for Sequence-to-Sequence Mapping

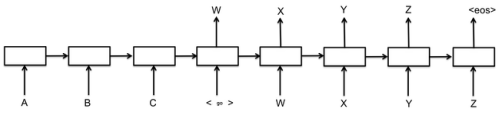
In user reviews, there may be a wide separation between product names, the feature under comparison and the user opinion. General RNNs are unable to capture such long term dependencies among words in practical scenarios.

Long Short Term Memory networks (LSTMs) are a special kind of RNN, capable of learning long-term dependencies. 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!

LSTMs have the concept of states and gates. State represents the amount of information a particular LSTM cell has saved from the past. Gates are used to modify the state and control what information should be forgotten (Forget Gate) and what information needs to be newly added (Input Gate).

9.2 Experimental Setup

- We limited the **length of any review sentence** fed to the LSTM Network to **50**.
- Now since, we have a one-to-one mapping from input sentence (sequence-1) to output labels (sequence-2), so the length of output sequence is also limited to 50.
- Each LSTM unit is responsible for one token/word of the sequence.
- Hence, we fixed the maximum size of each LSTM layer to 120, as per the diagram below.



- For experimental purposes, we fixed the number of layers in the network to 3.
- The model was trained on 290 review sentences and size of development set was fixed to 70 sentences. The model was run on department GPUs.

9.3 Observations

Since, our model has 3 layers of LSTM Units and 120 cells in each, so total number of weights to be trained is 360. For getting reasonable accuracy, we need a good large data-set that covers a diverse variety of scenarios and has at least 10 sentences per weight. Hence, our model is over-fitting with the provided set of sentences.

10 Future Work

We currently have approx. 2500 manually labeled sentences which include sentences labeled by us along with sentences from Jindal and Liu[1] and Kessler and Kuhn[4]. We would run structural alignment on the entire electronics corpus from Amazon, in parallel on the department CPUs and GPUs. This will give us a large labeled data-set.

Based on paper by Miwa and Bansal, 2016[8], we have a more sophisticated technique for entity labeling and relation classification in general sentences using Bi-directional LSTMs. This system has 3 layers. First, embeddings layer, where we supply vector representation for words, their POS Tags and Entity Labels. Then, in the second layer, we have sequentially arranged Bi-directional LSTM units to capture influence of contect words both before and after a target word before it is labeled. The third layer consists of tree-structured arrangement for LSTM cells which captures the dependency parse information in relation classification among the entities in a sentence.

We have currently used Tensorflow for LSTMs, but now, to make our system more efficient and for easy debugging, we would shift to Deep-Learning Toolkit developed by Microsoft Research called Computational Network Toolkit (CNTK).

11 References

1. N. Jindal , B. Liu. 2006. Identifying comparative sentences in text documents. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 244–251, Seattle, Washington, USA, August 11, 2006.

2. W. Kessler and J. Kuhn. 2013. Detection of product comparisons - how far does an out-of-the-box semantic role labeling system take you? In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1892-1897, Seattle, USA, October 2013.

3. W. Kessler and J. Kuhn. 2015. Structural alignment for comparison detection. In *Proceedings of the 10th Conference on Recent Advances in Natural Language Processing*, pages 275-281, Hissar, Bulgaria, September 2015.

4. W. Kessler and J. Kuhn. 2014. A Corpus of Comparisons in Product Reviews. In *Proceedings of the 9th Language Resources and Evaluation Conference*, pages 2242-2248, Reykjavik, Iceland, 28.-30. May 2014.

5. A. Björkelund et al. 2014. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, Boulder, Colorado, June 04, 2009.

6. B. Bohnet and J. Nivre. 2012. A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics*, pages 1455-1465, 2012.

7. B. Bohnet and J. Kuhn. 2012. The best of both worlds: a graph-based completion model for transition-based parsers. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 2012.

8. M. Miwa and M. Bansal. 2016. End-to-end Relation Extraction using LSTMs on Sequences and Tree Structures. In *the Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105-1116, 2016.