This file has been cleaned of potential threats.

If you confirm that the file is coming from a trusted source, you can send the following SHA-256 hash value to your admin for the original file.

7d39a6d83382d078285259b0c5441c47e2db0841ad71b189b049249ad9f38e31

To view the reconstructed contents, please SCROLL DOWN to next page.
Extracting Entities of Interest from Comparative Product Reviews

Jatin Arora  
Indian Institute of Technology Kharagpur  
jatinarora2702@gmail.com

Pawan Goyal  
Indian Institute of Technology Kharagpur  
pawang.iitk@gmail.com

Sumit Agrawal  
Indian Institute of Technology Kharagpur  
agrawal.sumit33@gmail.com

Sayan Pathak  
Microsoft Research, Redmond  
sayanpa@microsoft.com

ABSTRACT
This paper presents a deep learning based approach to extract product comparison information out of user reviews on various e-commerce websites. Any comparative product review has three major entities of information: the names of the products being compared, the user opinion (predicate) and the feature or aspect under comparison. All these informing entities are dependent on each other and bound by the rules of the language, in the review. We observe that their inter-dependencies can be captured well using LSTMs. We evaluate our system on existing manually labeled datasets and observe out-performance over the existing Semantic Role Labeling (SRL) framework popular for this task.

1 INTRODUCTION
User opinions have always had a strong influence on both producers and consumers in a market. In the past few years, with the advancement of e-commerce, a large proportion of these user opinions are present in the form of product reviews on online shopping websites like Amazon¹, Ebay² etc. Product specifications bring out only the quantitative aspects of the product, but consumers are often interested in the qualitative comparison among competing products. Manufacturers, on the other hand, read product reviews to know the market response for their products and top competitors currently in the market. But going through the large volume of reviews manually has become increasingly difficult. Hence, automated extraction of this product comparison information from raw reviews is a popular research area.

There can be various use cases for extracting information depending upon which, there can be variety of techniques to do the task. One can apply Named Entity Recognition (NER) to identify the products being compared and then do sentiment analysis to find the favored product. Sikchi et al. [11] use product specifications along with the review text for identifying the favored entity. Another way is to use text summarization to reduce the amount of text one has to manually read to infer the user opinion. Such techniques either do partial information extraction or require some manual intervention. We are interested in an automated full-scale extraction of comparison information from the reviews. In any review sentence involving comparison, there can be at most three major informing entities: the names of the two products being compared, the predicate or the user’s opinion and the feature (aspect) under comparison. Consider an example camera review given by a user, “Nikon Coolpix has better image quality than Cannon”. Given this review as input, we want to develop a system which can identify the products (“Nikon Coolpix”, “Cannon”), the aspect being compared (“image quality”) and the predicate or user opinion (“better”). A graphical representation of our system handling this example review is shown in Figure 1.

Kessler and Kuhn [6] model this as a Semantic Role Labeling (SRL) problem. In SRL, an event is expressed by the predicate (user opinion) and participants are the arguments that fill different semantic roles for the event. Here, the roles are the names of the products and the aspect being compared. They train a standard feature engineered SRL system [1] and show the best that can be achieved through it without major adaptations.

We observe that all these informing entities (predicate, aspect and product names) are dependent on each other and extraction of one is facilitated by the knowledge of the other entities. This motivates us to model the sentence as a whole, using Long Short Term Memory (LSTM) cells, which inherently capture the inter-dependencies among these informing entities. Through this work, we show how combined extraction of informing entities using deep learning outperforms the existing feature-engineered frameworks. We compare with two SRL baselines and evaluate the systems on two tasks, argument identification and argument classification, and obtain better F1-Scores in both the tasks.

¹https://www.amazon.com
²http://www.ebay.com

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference format:
https://doi.org/10.1145/3132847.3133141

KEYWORDS
Comparison Mining, Deep Learning, Opinion Extraction
2 DATASET

Full-scale annotation of informing entities in a comparison sentence is a difficult task due to the diversity of writing styles of the users. So, most existing annotated datasets in this domain are small and manually labeled. Since a deep learning framework generally requires training through a large number of samples, we combine the annotated data obtained from various existing sources and split it (60:40) for training and testing. In addition to this, we artificially annotate review sentences using a pattern matching technique (explained in the next section) and add these to the training set. We, then filter out and use only review sentences which have at-least one comparative predicate and have length less than or equal to 30. The manually labeled datasets used are explained below.

- **Jindal and Liu Corpus**: The corpus\(^3\) contains review sentences mostly of products in electronics domain, annotated and segregated into 4 comparison categories. This was used by Jindal and Liu \([3, 4]\). We use all comparison sentences from the corpus except type 4 (non-gradable comparisons). Each comparison sentence is annotated with names of the products (Entity 1 and 2), the aspect (Entity 3) and the predicate is mentioned as a bracketed comparison phrase.

- **Corpus by Kessler and Kuhn**: This corpus \([7]\) contains around 2200 manually annotated camera reviews. We use all the annotated sentences from here. The annotation scheme is the same as the one we use. Entities 1 and 2 are called products 1 and 2 in our nomenclature.

- **JDPA Corpus**: This corpus \([5]\) contains annotated blog posts containing user opinions about automobiles and digital cameras. We use only the sentences from the digital cameras domain which have the comparison class label in their annotation. The words marked by this class label bring out the user opinion and are marked as predicates. In addition, this class has 4 annotation slots, ‘More’, ‘Less’, ‘Dimension’ and ‘Same’. We map the ‘More’ slot to Product 1, ‘Less’ slot to Product 2, ‘Dimension’ slot to Aspect and ignore the ‘Same’ slot which indicates if the two products are ranked as equal.

- **Self Manual Labeling**: To include latest review trends, we crawled digital camera reviews from Amazon\(^4\), for the year 2016. Then, we manually annotated 350 review sentences with the three entities of information, wherever available. Overall contribution of different corpora in our training and test data is summarized in Table 1.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Train-Set</th>
<th>Test-Set</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>J&amp;L</td>
<td>313</td>
<td>208</td>
<td>521</td>
</tr>
<tr>
<td>Kessler</td>
<td>982</td>
<td>655</td>
<td>1637</td>
</tr>
<tr>
<td>JDPA</td>
<td>133</td>
<td>90</td>
<td>223</td>
</tr>
<tr>
<td>Manual</td>
<td>210</td>
<td>140</td>
<td>350</td>
</tr>
<tr>
<td>Pattern-Based</td>
<td>24164</td>
<td>0</td>
<td>24164</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>25802</td>
<td>1093</td>
<td>26895</td>
</tr>
</tbody>
</table>

\(^3\)Can be downloaded here, https://www.cs.uic.edu/~liub/FBS/data.tar.gz

\(^4\)https://www.amazon.com

3 PROPOSED APPROACH

3.1 Generation of Labeled Data

We observe that there are some distinct styles for expressing comparison in product reviews, generally used by people. Based on this observation, we made 5 simple patterns using regular expressions. If an unlabeled review sentence matches a pattern, we narrow down the exact regions to look for different entities of information, based on the pattern. The predicate is then identified by a comparative POS tag (JJR, JJS, RBR, RBS - as per the Penn Treebank Tagging scheme). The aspect and product names are identified by dictionary matching. The aspects dictionary has 83 features for products in the electronics domain. The products dictionary is 11,126 entries long. Both these dictionaries are made semi-automatically, i.e., first using some heuristics to get a list with good accuracy and then manually correcting it. As an example, consider the pattern, [Aspect] [Preposition (of | in)] [Product Name] [Opinion]. This pattern fits sentences like, “The zoom in Nikon S8100 is far better.” and labels zoom (Aspect), Nikon S8100 (Product1) and better (Predicate). These patterns certainly do not exhaustively capture all possible
comparisons, which is the final goal of this research work, but still give an annotated dataset with good precision, which can be used for training. We use this pattern fitting approach on electronic gadget reviews [8] from Amazon2. The labeled data hence generated is used in training only, as shown in Table 1.

3.2 Overall Framework
Our model consists of three layers. An input review sentence is first tokenized and then its words are embedded by passing through the embedding layer. The embedded sentence is then passed through a LSTM (Long Short Term Memory) layer, where corresponding to each word, we have one LSTM unit. For each word of the sentence, the output from the corresponding LSTM cell is converted to a 5-dimensional attribute vector by passing through a fully connected layer. The attribute vector has one dimension for each entity of information (Product1, Product2, Aspect, Predicate, None) and is converted to a probability distribution by passing through a softmax layer. Finally, we take the label for the word/token as the attribute having the maximum probability. An example review being processed by our model is shown pictorially in Figure 1.

3.3 Embeddings
For a word/token in a sentence, the embedding layer finds out two embeddings, the word embedding and the one-hot POS (Part of Speech) embedding and concatenates the two, to be fed to the LSTM layer. We use the universal POS tags for POS embedding. For word embeddings, we try out 100-dimensional, and the standard 300-dimensional GloVe [10] word embeddings trained on a general English corpus (Text8 Corpus6) and those trained specifically on electronics reviews from Amazon. We do not go for higher dimensional embeddings since that would increase the number of training parameters in our model and we may not be able to effectively train it using the current size of training data we have.

3.4 Training and Model Variants
We train our system to minimize the cross-entropy loss between the output probability distribution and the one-hot gold labels for tokens in sentences from the training set. There are several model variants that we test. We try out both unidirectional and bidirectional LSTMs. We work with both single and multiple LSTM layers. The specifications of all variants are shown in Table 2 and the results obtained by these variations are all reported in the next section. The model giving the best results is shown in bold (Model2).

4 EXPERIMENTAL RESULTS

4.1 Experimental Setup
For sentence and word tokenization as well as POS Tagging, we use Natural Language Toolkit (NLTK). The deep learning model implementation is done using Tensorflow. The embeddings are prepared using GloVe and are kept frozen, not trained with the main model. All the parameters of the model are randomly initialized. For baseline approaches, using Semantic Role Labeling (SRL), we use the same settings as used by Kessler and Kuhn [6]. The SRL system takes as input, data in CoNLL format for which we use the MATE7 Dependency Parser [2].

4.2 Evaluation Framework
We test our system as well as the baselines, using the manually labeled test data described in Table 1. We evaluate the systems on two tasks and in both cases, we calculate the Precision, Recall and F1-Scores. The first task is argument identification i.e. identifying if a word/token has some entity of information. The second task is, argument classification, where for a given word, the system has to classify it with one of the 5 labels (Predicate, Product 1, Product 2, Aspect, None).

4.3 Baseline Approaches
We compare our system with the approach presented in the paper by Kessler and Kuhn [6]. The SRL is a feature engineered machine learning based system. Their system uses standard SRL features for extracting all informing entities in a review using a 2-stage pipeline. It first identifies only the predicate using SRL. Then, in the second stage, uses predicate information (either gold labeled predicates, or those identified in the first stage) for identifying and classifying the other arguments. In their paper, the authors present their results using gold predicates and report a 10% decrease in the results if system identified predicates are used instead. We replicate their system and for a fair comparison with the proposed approach which is a single-stage model, we create two baselines. Baseline1: We use their method with the gold predicates information, and as mentioned in their paper, the results obtained from their system are decreased by 10% to compare with the proposed model. Baseline2: Instead of gold predicates, we feed in the system identified predicates from stage 1 of the pipeline to stage 2 of the SRL system and compare with our model’s performance. The results for predicate identification, argument identification and classification are shown in Tables 3, 4 and 5 respectively. Note that we do not show Baseline1 results in Table 3 as the gold standard predicates were used. We observe that a single layer of Bidirectional LSTMs, using 300 dimensional GloVe word embeddings prepared from general English (Text8) Corpus gives the best results overall and outperforms both baselines in all the tasks in terms of recall as well as F1-score8.

5 DISCUSSIONS
• Since a large amount of training data is generated using patterns, we observe a relatively low recall from the models trained using the data, as expected. But Baseline2 reports a very low recall. This

---

This corresponds to Model2, shown in Bold.

---


2.Can be downloaded from here, http://mattmahoney.net/dc/textdata

is because, in the pipelined SRL approach, correct identification of the predicate (the event) is key to further identification of arguments (roles). Since Baseline2 gives a high precision and very low recall for predicate identification itself on the test data, hence same is the trend for argument identification and classification as well. Our system, on the other hand, overcomes the limitation of a pipelined approach by combined modeling of the informing entities.

• Increasing the number of hidden LSTM layers does not improve the results, thus confirming that a single layer LSTM rightly captures the dependencies among the informing entities in a comparison based review sentence.

• Using 100 dimensional word embeddings leads to a lower recall. But, since the embedding dimensions are proportional to trainable model parameters, smaller dimensional embeddings can give a good enough model even when the training set is small.

• Embeddings specifically prepared from the electronics corpus give a slightly better precision but compromise with the recall. Hence, general English text embeddings and electronics embeddings both give almost similar F1-score on both tasks.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a simple framework which uses deep learning to annotate and hence, extract all important entities of information from comparative product reviews. This system saves the trouble of feature engineering and gives better results than the previously presented SRL based system.

We also developed simple patterns which capture some common styles of presenting comparisons in reviews. This pattern fitting technique proved beneficial in expanding our training data, making it possible for the deep learning model to effectively learn the sentential structure and inter-dependencies among the informing entities in comparative reviews.

There is still a lot of scope for improvement. In reviews, users often tend to use pronouns or refer implicitly to a product mentioned in the previous sentences. In such cases, a wrapper system needs to be developed which can capture the sentence-to-sentence dependencies and map the pronoun in the current sentence, to the corresponding noun mentioned in the previous sentences. This is an active area of research which we would like to explore. Peng et al. [9] show the effectiveness of Graph LSTMs for such cross-sentence relation extraction.

ACKNOWLEDGMENTS

The authors would like to acknowledge the contributions of Yash Agrawal and Kushagra Aggarwal, CSE, IIT Kharagpur, in parsing the JDPA Corpus and manual annotation of datasets.

REFERENCES