Towards sentiment augmented predictive techniques in natural language

Thesis submitted in partial fulfillment of the requirements for the degree of

*Master of Science*

in

*Computer Science and Engineering by Research*

by

Battu Varshit

201402029

battu.varshit@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
July 2019
CERTIFICATE

It is certified that the work contained in this thesis, titled “Towards sentiment augmented predictive techniques in natural language” by Battu Varshit, has been carried out under my supervision and is not submitted elsewhere for a degree.

__________________________  __________________________
  Date                       Advisor: Prof. Radhika Mamidi
To all my well wishers
I would like to take this opportunity and thank all my well wishers for supporting me throughout my life at IIIT Hyderabad. This thesis is the final result of great expertise, support and motivation from not one but many people be it professors, friends, wing-mates.

Firstly, I would like to thank my parents and sister for their continuous support and encouragement. They helped me walk through the most difficult times. I was able to pursue my research with passion due to their endless love and guidance.

I am deeply indebted to Prof. Radhika Mamidi for being my advisor. She helped me understand what research truly is. She allowed me to choose from topics within my interest so that I can work with passion. Me being able to graduate in 5 years is the result of her helping me in every step of the way. Her guidance helped me finish my research and write my thesis. She makes the research meetings seem informal. One of the best things is that she understands the way a student thinks.

Life in college wouldn’t have been the same without my friends. Thank you Vishal, for being my roomie and keeping up with all the high level stuff I did and for your enlightening thoughts. Thank you Harsha, Ram and Vivek for the late night discussions on life. Thank you Murali for being a part of my research and helping me out. Thank you Navyasri, Sreeja and Pratyusha for keeping my hopes up during desperate times and for the high level memories we made. Thank you Rama Rohit, for helping me finish up a paper during my internship. A big thanks to all my wing-mates Yashwanth, Rohith, Hareesh, Mahesh, Rishith, Veerendra for their innovative opinions and discussions which helped me in both my research and life.

Finally, I would like to thank God for giving me all the strength, knowledge, courage and loving people around me to help me finish my work.
Abstract

Sentiment is an important feature of any text. Feelings and opinions are purely subjective, unlike facts. Analyzing these opinions accurately is a challenging task. However, using the sentiment, we are looking to understand the attitude of a writer with regard to a specific topic in a piece of text. Sentiment analysis has been an important topic of research since a very long time. Simply put, the task involves a system that would be predicting (classifying) the sentiment of a given input sentence as either positive, neutral or negative. Sometimes we have more fine-grained classes and sometimes we even evict the neutral class based on what we need to use the sentiment for. The capability to automatically compute the sentiment for things such as user reviews, critique on movies, e-commerce etc is of tremendous importance. It helps organizations and businesses take multiple large scale important decisions which are based strongly on user satisfaction.

Movies are one of the most prominent means of entertainment. The favourite pastime of many people would be watching movies. They provide a break from the hectic schedule a person goes throughout the day. People often prefer to express their views online in English as compared to other local languages because of many factors such as non availability of applications which makes typing in local languages easy whereas typing in English is very easy. Even if one manages to write a review in a local language there are very less number of platforms which accept such reviews. This leaves us with a very little amount of data in languages apart from English to work on.

The widespread use of the Internet in recent times has led to large volumes of data related to movies being generated and shared online. People watch movies, write reviews and give ratings online. This method of broadcasting opinions has gained a lot of popularity ever since. However, this led to a decrease in the quality of opinions that were shared. Due to this, people find it challenging and difficult to browse through all the opinions. This issue of bogus and random opinions is witnessed in a lot of cases where the user can provide feedback in a quick manner such as multiple choice options, checkboxes etc. Movie ratings and genres fall in this category. Movie ratings and genres play an important role in tasks such as user movie recommendations, verifying the relationship between user-submitted reviews and ratings etc. The ability to predict the correct rating/genre of a movie would be useful considering these aspects.

In this thesis, we attempt to solve problems we face in real life with the help of sentiment. One problem is encountering invalid information(wrong ratings) by using the sentiment of the movie reviews. We propose methods to predict the movie rating based on its summary. We then set out to use priors that are
generally available with movie summaries in order to improve the accuracy. In order to achieve this, we consider the associated movie reviews as well while predicting the rating and provide insights on why this helps our models perform better. We use sentiment of reviews along with the summary in order to predict the rating more accurately since the sentiment captures a lot of essential information that can aid rating prediction. The majority of methods used to study NLP problems employed shallow machine learning models and time-consuming, hand-crafted features for a very long time. Many problems were encountered due to this. One of the problem is the curse of dimensionality as linguistic information was represented with sparse representations. However, with the recent popularity and success of word embeddings, sentence embeddings, neural network based models have achieved better results on various language-related tasks as compared to traditional machine learning models like SVM or logistic regression. Deep learning methods are starting to out-compete the statistical methods on some challenging natural language processing problems with singular and simpler models. We propose deep learning models to solve our problem at hand and compare the results with traditional methods. The other problem is preserving sentiment during translation. An important aspect of translation is ensuring that the complete meaning (including the sentiment/opinion) of the source text is translated appropriately. Consider product reviews, retaining the sentiment is an important aspect of translating these reviews. We cannot afford to have phrases such as “not at all good” get translated to “not good” since there is a significant difference in the sentiment of the two phrases. Most of the metrics that evaluate machine translators only consider n-grams, number of overlaps, etc in order to produce translation scores. This does not account for the preservation of sentiment. In order to solve this other problem, we propose a new metric that considers the sentiment of a sentence along with the existing means of evaluation.
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 What is sentiment?</td>
<td>2</td>
</tr>
<tr>
<td>1.2 How is sentiment calculated?</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Traditional methods vs Deep Learning</td>
<td>3</td>
</tr>
<tr>
<td>1.4 About Telugu Language</td>
<td>3</td>
</tr>
<tr>
<td>1.5 Motivation</td>
<td>4</td>
</tr>
<tr>
<td>1.6 Contribution of this thesis</td>
<td>4</td>
</tr>
<tr>
<td>1.7 Thesis Organisation</td>
<td>5</td>
</tr>
<tr>
<td>Related work</td>
<td>6</td>
</tr>
<tr>
<td>Rating prediction using multimodal information</td>
<td>10</td>
</tr>
<tr>
<td>3.1 Background</td>
<td>10</td>
</tr>
<tr>
<td>3.2 Dataset</td>
<td>11</td>
</tr>
<tr>
<td>3.2.1 Preprocessing</td>
<td>12</td>
</tr>
<tr>
<td>3.2.2 Statistics</td>
<td>12</td>
</tr>
<tr>
<td>3.3 Score Prediction</td>
<td>13</td>
</tr>
<tr>
<td>3.3.1 Trailers</td>
<td>14</td>
</tr>
<tr>
<td>3.3.2 Summaries</td>
<td>14</td>
</tr>
<tr>
<td>3.3.3 Method</td>
<td>15</td>
</tr>
<tr>
<td>3.3.4 Implementation Details</td>
<td>16</td>
</tr>
<tr>
<td>3.4 Results</td>
<td>16</td>
</tr>
<tr>
<td>3.4.1 Qualitative Analysis</td>
<td>16</td>
</tr>
<tr>
<td>3.4.2 Empirical Validation</td>
<td>17</td>
</tr>
<tr>
<td>3.5 Summary and Future Work</td>
<td>18</td>
</tr>
<tr>
<td>Predicting the Genre and Rating of a Movie Based on its Synopsis</td>
<td>19</td>
</tr>
<tr>
<td>4.1 Background</td>
<td>19</td>
</tr>
<tr>
<td>4.2 Dataset</td>
<td>20</td>
</tr>
<tr>
<td>4.2.1 Data Extraction</td>
<td>21</td>
</tr>
<tr>
<td>4.2.2 Preprocessing</td>
<td>21</td>
</tr>
<tr>
<td>4.2.2.1 Grouping of genres</td>
<td>21</td>
</tr>
<tr>
<td>4.2.3 Statistics</td>
<td>21</td>
</tr>
<tr>
<td>4.3 Genre and Rating Prediction</td>
<td>22</td>
</tr>
<tr>
<td>4.3.1 Character Embeddings</td>
<td>22</td>
</tr>
</tbody>
</table>
## CONTENTS

4.3.2 Word Embeddings ........................................... 23
4.3.3 Sentence Embeddings ...................................... 23
4.3.4 Concatenated Embeddings ................................. 24

4.4 Experiments ....................................................... 24
4.4.1 Experimental Details ..................................... 24
4.4.1.1 SVM ......................................................... 25
4.4.1.2 Random Forests ....................................... 25
4.4.1.3 CNN ......................................................... 25
4.4.1.4 LSTM ....................................................... 25
4.4.1.5 GRU ......................................................... 25
4.4.1.6 FCNN ...................................................... 25
4.4.1.7 Hybrid Model ........................................... 26
4.4.2 Results ......................................................... 27
4.4.3 Analysis ....................................................... 27

4.5 Summary ........................................................ 28

5 Sentiment as a prior for movie rating prediction ............... 29
5.1 Background ...................................................... 29
5.2 Dataset .......................................................... 30
5.2.1 Analysis of the Dataset .................................... 30
5.2.2 Dataset statistics .......................................... 31
5.3 Sentiment prediction .......................................... 31
5.3.1 Recurrent networks ....................................... 31
5.3.2 Convolution networks ................................... 32
5.4 Rating prediction ............................................... 32
5.4.1 Prediction without priors .................................. 32
5.4.1.1 Character embeddings ................................ 33
5.4.1.2 Word embeddings ..................................... 33
5.4.1.3 Sentence embeddings ................................ 33
5.4.2 Prediction with sentiment priors ......................... 34
5.4.2.1 Inclusion of priors ..................................... 34
5.4.3 Empirical validation and explanation .................... 34
5.5 Experiments ...................................................... 35
5.5.1 Results ....................................................... 35
5.5.1.1 Analysis .................................................. 35
5.6 Summary ........................................................ 36

6 SEMMT: Sentiment based evaluation Metric for Machine Translation ........................................ 37
6.1 Background ...................................................... 37
6.2 Popular metrics ............................................... 37
6.2.1 BLEU ......................................................... 38
6.2.2 CIDEr ......................................................... 38
6.2.3 METEOR ...................................................... 38
6.2.4 ROUGE ....................................................... 38
6.3 Dataset .......................................................... 38
6.4 Sentiment ........................................................ 39
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 An overview of our pipeline corresponding to Model-1. We start with the trailers and summaries as inputs and predict the G-Score classes as outputs. The Inception-V3 pre-trained model is used to extract features of video frames. ConvPoolBlocks and ConvBlocks are described in Figure 3.2. The output sizes at each of the layers are mentioned in the figure.</td>
<td>13</td>
</tr>
<tr>
<td>3.2 ConvPoolBlocks and ConvBlocks are used in order to process the summaries in our proposed models. A ConvPoolBlock consist of Convolution, Tanh, MaxPool and Dropout layers. A ConvBlock consists of Convolution, Tanh and Dropout layers.</td>
<td>15</td>
</tr>
<tr>
<td>3.3 Qualitative examples where the model predicts the correct G-Score class for Super Mario Odyssey (left) and the incorrect G-Score class for Gran Turismo 6 (right). The true G-Score for Super Mario Odyssey is 93 and the true G-Score for Gran Turismo 6 is 81.</td>
<td>17</td>
</tr>
<tr>
<td>4.1 Branched CNN based genre/rating prediction model architecture with character inputs which considers the synopsis to perform the prediction. The Conv blocks represented in the figure consist of Convolution, ReLU and Max-pool layers. We used Dropouts at various places for regularization. Word inputs are similar except that the input size is different.</td>
<td>23</td>
</tr>
<tr>
<td>4.2 LSTM based genre/rating prediction model architecture with character inputs which considers the synopsis to perform the predictions. Replacing LSTM cells with GRUs would give us GRU models.</td>
<td>24</td>
</tr>
<tr>
<td>4.3 Hybrid model based on stacking three models to predict genre/rating with word, character and sentence embeddings as the input considers synopsis to perform the prediction. We used Dropouts at various places for regularization.</td>
<td>26</td>
</tr>
<tr>
<td>5.1 Branched CNN based rating prediction model architecture which considers sentiment priors along with the summaries to predict the ratings. The Conv blocks represented in the figure consist of Convolution, ReLU and Max-pool layers. We used Dropouts at various places for regularization.</td>
<td>32</td>
</tr>
<tr>
<td>5.2 LSTM based rating prediction model architecture which considers sentiment priors along with the summaries to predict movie ratings.</td>
<td>33</td>
</tr>
<tr>
<td>6.1 Graph of the proposed SEMMT formulation.</td>
<td>42</td>
</tr>
</tbody>
</table>
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Distribution of VGD according to Genre classes, G-Score classes and Age Ratings</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Our proposed grouping of genres into 5 classes based on similarity</td>
<td>12</td>
</tr>
<tr>
<td>3.3 Results on predicting G-Scores using 10-fold cross-validation. We present mean accuracies on all three of our models considering trailers only as inputs as well as both trailers and summaries as inputs. Improvements obtained with each of the models are also mentioned.</td>
<td>18</td>
</tr>
<tr>
<td>4.1 Number of data-points per genre/rating for each language. Not all languages have data belonging to all classes, the classes not corresponding to a language are marked (in red)</td>
<td>20</td>
</tr>
<tr>
<td>4.2 Preprocessing the genres to form 9 genre classes that are used for genre prediction.</td>
<td>22</td>
</tr>
<tr>
<td>4.3 Size of the dataset created</td>
<td>22</td>
</tr>
<tr>
<td>4.4 Genre prediction accuracies for various models on each of the languages in MLMRD</td>
<td>26</td>
</tr>
<tr>
<td>4.5 Rating prediction accuracies for various models on each of the languages in MLMRD</td>
<td>27</td>
</tr>
<tr>
<td>5.1 Dataset statistics for both the sentiment classification and the rating prediction datasets created. Sentiment classification dataset consists of reviews whereas rating prediction dataset consists of reviews and summaries together</td>
<td>30</td>
</tr>
<tr>
<td>5.2 Count of movies per rating class present in the dataset</td>
<td>30</td>
</tr>
<tr>
<td>5.3 Time-span of movies released per year present in the dataset</td>
<td>31</td>
</tr>
<tr>
<td>5.4 Sentiment classification experiments with various models on 50-50 and 80-20 train-test data splits.</td>
<td>31</td>
</tr>
<tr>
<td>5.5 Results on rating prediction for movies. Considering an 80-20 train-test split</td>
<td>35</td>
</tr>
<tr>
<td>6.1 Increase in values after considering sentiment</td>
<td>44</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

As the amount of data present online increases exponentially day by day, we have reached a point where a human cannot comprehend all of it in a meaningful manner due to its sheer size. This lead to work on automated recommender systems. The main issue with these kinds of methods is that not all the information is present online and all the information present need not be correct. Automated movie genre and rating prediction have a lot of applications. We can recommend same genre movies based on one’s previous watch history. Genre of a movie can be identified by its synopsis. Recommending a movie only based on its genre is not a good idea as the same genre can have both good and bad movies. So recommending movies based on both genre and rating would result in a proper recommendation system. But the main problem here is that people do not often tend to rate the movie they watch, thus automated rating prediction would be of great help for recommendation systems. Though the rating of a movie depends on multiple factors like actors, screenplay, direction etc. but that information is very difficult to capture through the available data. In most of the cases, the synopsis of the movie plays a crucial impact on audience rating.

There is very little amount of data in languages apart from English to work on. To overcome this, we created the Multi-Language Movie Review Dataset (MLMRD). The dataset consists of genre, rating, and synopsis of a movie across multiple languages, namely Hindi, Telugu, Tamil, Malayalam, Korean, French, and Japanese. The balance in the dataset is not as expected because nowadays movies in specific languages tend to belong to only specific genres due to various reasons like movie collections, ease of making etc. For example, no documentary movies are present in Telugu as such movies make fewer collections at Tollywood box office.

We first propose multiple deep-learning based methods to predict the genre and rating of a movie based on its synopsis. The sentiment of reviews is a valuable feature that we use in order to solve the task at hand better. Using this feature would lead to more accurate rating predictions and would help people make select the right movie to watch. Later we propose a solution to predict the rating for a movie considering the features embedded in the reviews along with the summaries. Traditional methods such as support vector machines (SVMs) and logistic regression do not perform well at the given task and hence we propose a deep learning based method. We verified and validated our method
on Telugu movie reviews. Telugu is a Dravidian language for which very few resources and tools are available. It is morphologically very rich agglutinative language which makes it hard to perform any natural language processing related tasks on it with “perfect” tools being unavailable. Then we move on to propose a metric which uses sentiment to give a SEMMT score. We can use the score to evaluate machine translators. We try to overcome the problems faced by the other metrics available and consider sentiment at the same time.

1.1 What is sentiment?

The way in which humans speak is really complex. We can guess the mood a person is in by looking at the person’s choice of words be it during face to face conversations or during texting even if the intention is not explicitly shown. Sentiment can be seen as a mixture of how someone is feeling, how they are wording their sentences etc. It is broadly divided into 3 classes namely positive, neutral and negative. The method of analysing the sentiment is called sentiment analysis or opinion mining. Systems which perform sentiment analysis mainly look for three things - sentiment, subject and the opinion holder. With the help of these systems, this unstructured information could be easily and automatically transformed into structured data of public opinions about products, services, brands, politics, or any topic that people can express opinions about. This data can be very useful for commercial applications like marketing analysis, public relations, product reviews, net promoter scoring, product feedback, and customer service. In a world where we generate 2.5 quintillion bytes of data every day, sentiment analysis has become a key tool for making sense of that data. This has allowed companies to get key insights and automate all kind of processes.

1.2 How is sentiment calculated?

There are many methods and algorithms to calculate the sentiment. They can be coarsely classified into three types

- Rule-based : Manually defined rules give a sentiment score.
- Automatic : Machine learning techniques are used to predict the sentiment score.
- Hybrid : Both the above approaches are combined and the sentiment score is calculated.

Rule-based approach is a very basic method as it doesn’t take into account the sequence of words. A more advanced processing can be made, but these systems get very complex quickly. They can be very hard to maintain as new rules may be needed every time a for new expression and vocabulary comes up. Besides, adding new rules may have undesired outcomes as a result of the interaction with previous rules. As a result, these systems require important investments in manual fine tuning and maintaining the rules.
Automatic approach makes use of machine learning techniques. The sentiment analysis task is usu-
ally modeled as a classification problem where a classifier is fed with a text and returns the correspond-
ing category, e.g. positive, negative or neutral. A large annotated corpus is given to the machine learning 
model. It trains on the data, learns patterns and predicts the sentiment score of a new input.

In this thesis, we make use of machine learning techniques to predict the sentiment and rating. In the 
coming chapters, we discuss how and why these techniques give good results.

1.3 Traditional methods vs Deep Learning

Deep learning is a diverse set of algorithms that attempts to imitate how the human brain works by 
employing artificial neural networks to process data. In traditional Machine learning techniques, most 
of the applied features need to be identified by a domain expert in order to reduce the complexity of 
the data and make patterns more visible to learning algorithms to work. The biggest advantage of Deep 
Learning algorithms are that they try to learn high-level features from data in an incremental manner. 
This eliminates the need of domain expertise and hard core feature extraction. We believe that one 
of the main reasons for this is that deep learning based approaches tend to generalize a lot better as 
compared to traditional methods and hence they perform a lot better on unseen test data. Due to this 
there is a huge difference in the performance of traditional machine learning approaches such as SVMs 
and Random Forests when compared to deep learning based methods such as FCNNs. They all use 
the same inputs which are the embeddings generated using the Doc2Vec model. We have also tried to 
see how these methods perform if the testing data is just a subset of the training data. In this case, we 
notice that they are both able to represent their training data well and hence achieve similar accuracies 
during testing. However, this only happens since the number of data points in our dataset are not many. 
When the number of data points increases, deep learning based approaches show tremendous amounts 
of generalizability which allows them to attain much higher accuracies compared to traditional methods.

1.4 About Telugu Language

Russian linguist M.S. Andronov stated that Proto-Dravidian gave rise to 21 Dravidian Languages 
which can be broadly classified into three groups namely Northern group, Central group and Southern 
group of Dravidian languages. The Northern Group consists of three languages, the Central Group 
consists of ten languages and the Southern Group consists of seven languages. Out of the ten languages 
in the Central Group only Telugu became a language. Telugu split from Proto-Dravidian between 1500 
BC and 1000 BC. Hence, Telugu became a distinct language by the time any literary activity began to 
appear in the other languages. Kannada is Telugu’s closest cousin. Telugu is the most widely spoken 
language amongst those languages which use the Brahmi script. Bhadriraju Krishnamurti [55] speaks 
about the phonological and grammatical structure of the whole Dravidian family from different aspects 
in his book “The Dravidian Languages”. He also describes its history and writing system, discusses its
structure and typology, and considers it’s lexicon. Distant and more recent contacts between Dravidian and other language groups are also discussed.

1.5 Motivation

There are only a few languages such as English, German, Chinese, French, Spanish etc which receive a lot of attention from people who research in NLP. Due to this many resources like POS taggers, Tree-banks, Senti-WordNets etc are available in those languages. The NLP techniques developed for these languages cannot be used for low resourced languages as they are made to fit too strongly, on a huge dataset with various features. Using these on small datasets would lead to very poor performance. Hence, there is a huge need to work on resource constrained languages. Research into language-independent NLP methods that are appropriate in low-resource settings is desperately needed as such techniques can be applied to many low-resource languages at once. Telugu is one such low resourced language and my mother tongue. Besides this, there are many problems in natural language which we come across in real life where taking sentiment into consideration can provide a huge boost in the results we can achieve. The above reasons motivated me to work on Telugu and tackle the problems mentioned earlier using sentiment.

1.6 Contribution of this thesis

- Predicting the rating of video games using multimodal information
  - We collect trailers and their summaries.
  - We propose a method to pick certain frames from the trailer.
  - We use the inception model V3 to extract features from the trailers.
  - We then extract features from the summaries.
  - We perform multiple experiments and validate our claims by doing a 10-fold cross validation.

- Predicting the Genre and Rating of a Movie Based on its Synopsis
  - We collect synopsis, rating and genre of movies for 6 different languages.
  - We propose novel deep learning methods to predict rating and genres.
  - We also propose a novel ensemble model which utilizes word, char and sentence embeddings to predict the rating.
  - We perform multiple experiments and show that the proposed models give a superior increase in accuracy when compared to the traditional methods.
- Predicting the rating of a movie by using sentiment as a prior
  - We collect synopsis and ratings of movies in Telugu.
  - We propose novel deep learning methods to predict rating.
  - We use the proposed models by taking sentiment as a feature and predict rating.
  - We show an increase of 2% in the accuracy after taking sentiment as a feature.

- Providing a metric called SEMMT to evaluate machine translators using sentiment
  - We propose a novel metric which uses sentiment to give a score to machine translators.
  - We use skip thought cosine similarity as a base score for the proposed metric.
  - We show an increase of 41.2% in the correlation with human scores after considering sentiment as a feature.

The major contribution of this thesis is that we provide resources, generalized architectures and formulae which use sentiment as a feature to solve problems in natural language. Links to all the codes and datasets are mentioned in the respective chapters.

1.7 Thesis Organisation

The thesis is organised into seven chapters. Chapter 2 provides all the works which are relevant to the problem at hand. Chapter 3 speaks about how multimodal information can be used to boost the rating prediction accuracy. Here multimodal refers to images and text. Multiple experiments have been carried out and the best ones have been put here. Chapter 4 sheds light upon the type of generalized models that can predict rating and genre for different languages. We present novel architectures which can be used for rating prediction. Chapter 5 talks about how sentiment can be used to increase the prediction accuracy. Chapter 6 talks about how preserving sentiment during translation is important. It also provides a metric which uses sentiment as a feature to score a machine translator. Every chapter provides comparisons of our novel models with the baseline models in order to validate our claims. Each chapter is like a markov process i.e it doesn’t depend on the previous chapters. This is to ensure that the readers can read any chapter which they find interesting directly.
Chapter 2

Related work

There have been multiple works in the areas related to video and text classification, however, they often deal with domain specific information. Nominal work has been done on video game trailers in the past. We use video game trailers along with reviews to perform a cross-domain analysis in order to predict ratings. Zhang et al. [68] propose a supervised learning technique for summarizing videos by automatically selecting key-frames. They use Long-Short-Term Memory to model the variable-range temporal dependency among frames so that both representative and compact video summaries can be generated. Venugopalan et al. [63] look into how linguistic knowledge taken from large text corpus can aid the generation of natural language descriptions of videos. Haninger et al. [20] quantified and characterized the content in video games rated T (for "Teen") and measure how accurate the ESRB-assigned content descriptors displayed on the game box are to the real game. Simonyan et al. [54] investigate architectures of discriminatively trained deep Convolutional Networks for action recognition in videos. Capturing the complementary information from still frames and motion between frames was a challenge they address. Kahou et al. [26] present an approach to learn several specialist models using deep learning techniques. Among these are a convolutional neural network focusing on capturing visual information in detected faces, a deep belief net which focuses on the representation of the audio stream, a K-Means based "bag-of-mouths" model, which extracts visual features around the mouth region and a relational auto-encoder, which addresses spatiotemporal aspects of videos. Le et al. [34] present unsupervised feature learning as a way to learn features directly from video data. They presented an extension to the Independent Subspace Analysis algorithm to learn invariant spatiotemporal features from unlabeled video data. Zhou et al. [73] formalize multi-instance multi-label learning in which each training example is associated with not only multiple instances but also multiple class labels. They propose algorithms for scene classification based on the relationship between multi-instance and multi-label learning.

Glorot et al. [19] propose a deep learning approach that learns to extract a meaningful representation for each review in an unsupervised manner. Sentiment classifiers trained with this high-level feature representation clearly outperform state-of-the-art methods. Zhang et al. [70] show empirical exploration on the use of character-level convolutional networks (ConvNets) for text classification. They built large-
scale datasets to show that character-level convolutional networks can achieve state-of-the-art results. Iyyer et al. [24] present a simple deep neural network that competes with and sometimes outperforms models on sentiment analysis and factoid question answering tasks by taking only a fraction of the training time. Baker et al. [5] describe the application of Distributional Clustering to document classification. Their approach clusters words into groups based on the distribution of class labels of each word. Unlike techniques such as Latent Semantic Indexing, they were able to compress the feature space, while maintaining the classification accuracy. Poria et al. [47] use the extracted features in multimodal sentiment analysis of short video clips representing one sentence each. They use the combined feature vectors of textual, visual, and audio modalities to train a classifier which is based on multiple kernel learning, which is known to be good at heterogeneous data. Zhang et al. [69] mention that this learning problem is addressed by using a method called M\textsuperscript{LNB} which adapts the traditional naive Bayes classifiers to deal with multi-label instances. Feature selection mechanisms are incorporated into M\textsuperscript{LNB} to improve its performance.

Basu et al. [7] propose an inductive learning approach to predict user preferences. Huang et al. [22] propose a movie genre classification system using a meta-heuristic optimization algorithm called Self-Adaptive Harmony Search. Rasheed et al. [49] present a method to classify movies on the basis of audio-visual cues present in previews which contain important information about the movie. Zhou et al. [72] present a method for movie genre categorization of movie trailers, based on scene categorization. Gabriel S. Simoes et al. [53] explored CNNs in the context of movie trailers genre classification. Firstly, a novel movie trailers dataset with more than 3500 trailers was publicly released. Secondly, a novel classification method was done which encapsulates a CNN architecture to perform movie trailer genre classification, namely CNN-MoTion. Chin-Chia Michael Yeh et al. [66] concerns the development of a music codebook for summarizing local feature descriptors computed over time. With the new supervised dictionary learning algorithm and the optimal settings inferred from the performance study, they achieved the state-of-the-art accuracy of music genre classification. Aida Austin et al. [4] created a database of film scores from 98 movies containing instrumental (non-vocal) music from 25 romance, 25 drama, 23 horror, and 25 action movies. Both pair-wise genre classification and classification with all four genres was performed using support vector machines (SVM) in a ten-fold cross-validation test. Jnatas Wehrmann et al. [64] talked about a novel deep neural architecture based on convolutional neural networks (ConvNets) for performing multi-label movie-trailer genre classification. It encapsulates an ultra-deep ConvNet with residual connections, and it makes use of a special convolutional layer to extract temporal information from image-based features prior to performing the mapping of movie trailers to genres. Yong-Bae Lee et al. [36] presented a method for automatic genre classification that is based on statistically selected features obtained from both subject-classified and genre-classified training data. Pouya Ghaemmaghami et al. [18] addressed the specific problem of genre classification of movie clips using magnetoencephalography (MEG) data. They used the correlation analysis to show that genre related information is present in the visual and temporal areas of the brain and how these genre related brain signals can be decoded to target genre classes using the brain decoding paradigm. Junyong You
et al. [67] presented a semantic framework for weakly supervised video genre classification and event analysis jointly by using probabilistic models for MPEG video streams.

Movie rating prediction has been an important task for a long time and a lot of people have worked on it. Li et al. [37] use a regression model by incorporating reviewer and product information to predict ratings of movies. Lim et al. [38] use a variational Bayesian approach to predict movie ratings, they solved the problem of data over-fitting by using the SVD algorithm. Armstrong et al. [3] discuss the prediction of movie ratings after learning the relationship between the rating and a movie's various attributes using a training set based on kernel regression and model trees. Fikir et al. [16] use collaborative filtering based methods to predict ratings. Fikir et al. [15] also propose a method that uses simple matrix based factorization on the Netflix movie rating prediction dataset. Ganu et al. [17] present two new ad-hoc and regression-based recommendation measures, both of which take into account the textual component of user reviews. Nair et al. [45] present a rule-based approach for sentiment analysis from Malayalam movie reviews. Taboada et al. [60] use an approach based on lexicons to find the sentiments of sentences. Prerana Singhal and Pushpak Bhattacharyya. [1] conducted a survey on the performance of various neural architectures on sentiment analysis. Kim et al. [29] report a series of experiments using Convolutional Neural Networks for sentence classification. Wilson et al. [65] discuss phrase-level sentiment analysis, through which they identify the contextual polarity for a lexicon of sentiment expression.

Mullen et al. [44] conducted experiments on sentiment analysis using support vector machines (SVM) by adding new information sources as features which previously used the limited bag of words approach. Zhang et al. [71] propose a different type of character-based Convolutional Networks for text classification. Kennedy et al. [28] presents two methods for determining the sentiment expressed by a movie review. Socher et al. [56] discuss using the parse tree of the sentence to train RTNT(Recursive Neural Tensor Network) which performs better than the basic bag-of-words approach. Maas et al. [42] demonstrate the generation of the word vectors capturing sentiment which enhances the accuracies of deep learning techniques. Lakkaraju et al. [31] have tried aspect level sentiment analysis using hierarchical Deep learning techniques. Bespalov et al. [8] propose projection of higher order sentences to low dimensional space, over which a classifier can be used to predict the sentiment of the sentence.

The issue of not being able to capture the sentiment well in translation has been discussed in the past. [41] mentions how most of the common translation methods that exist currently often cause the sentiment encoded in the source sentence to be altered when we arrive at the translation. However, they try to attain higher translation quality instead of sentiment transfer. One issue that they have is that sentiment in itself cannot be strictly classified as it is a continuous feature and not discrete. In our work, we aim to target this issue and work on how the true nature of sentiment can be captured in translators. They also mention that they feel a small drop in translation quality is acceptable as long as the sentiment is transferred properly for certain use cases.

There has also been work done in the past which deals with calculating the sentiment of sentences by translating the sentence to a common language and then calculating the sentiment over there. An example of this is where Arabic was translated to English in order to calculate the sentiment since sentiment analyzers in English are more widely used and are in abundance as mentioned by [43] in their work. [2] also reveal that the simple translation of non-English texts into English and then evaluating the sentiment often performs better than the existing state of the art language-specific methods. These use cases strengthen the point as to why maintaining the sentiment while translation is of utmost importance in a lot of tasks and our work would be aiming to do exactly this.

[40] show how sentiment preservation in machine translation of user-generated content is important and how common it is for traditional translation systems to end up reversing the sentiment completely let alone alter it. They propose four methods in order to evaluate this with their augmentation of translation models based on a phrase-table fill-up method being the most significant at maintaining machine translation quality while improving sentiment preservation. They also show that using a translation system that conveys a different sentiment than the source can worsen the translation quality as well along with sentiment preservation in a lot of scenarios.
Chapter 3

Rating prediction using multimodal information

3.1 Background

Video games have become an integral part of most peoples lives in the recent times. This led to an abundance of data related to video games being shared online. However, this comes with issues such as incorrect ratings, reviews or anything that is being shared. Recommendation systems are powerful tools that help users by providing them with meaningful recommendations. A straightforward approach would be to predict the scores of video games based on other information related to the game. It could be used as a means to validate user-submitted ratings as well as provide recommendations. This work provides a method to predict the G-Score, that defines how good a video game is, from its trailer (video) and summary (text). We first propose models to predict the G-Score based on the trailer alone (unimodal). Later on, we show that considering information from multiple modalities helps the models perform better compared to using information from videos alone. Since we couldnt find any suitable multimodal video game dataset, we created our own dataset named VGD (Video Game Dataset) and provide it along with this work. The approach mentioned here can be generalized to other multimodal datasets such as movie trailers and summaries etc. Towards the end, we talk about the shortcomings of the work and some methods to overcome them.

Video games are almost everywhere these days, from individual consumers who play video games for fun to serious E-Sports professionals. The video game industry is a billion dollar industry. It was valued at $44.9 billion back in 2007 which rose to $91.5 billion in 2015. The increase in the rate of development of games and the number of people who play these games spiked up hand in hand throughout the world over the recent years. This increase in the sheer number of games marketed required people to rely on a trusted resource that would give them information about these games since it is not feasible for a human to keep the details of every single game ever released in memory. Another trend observed in recent times is that there is an exponential increase in the amount of data shared online. This, however, comes with certain unforeseen consequences such as a reduction in the quality of data present online, the spread of bogus information i.e false information being shared online. Considering the video game industry, people often rely on various sites to provide them with ratings, reviews etc of games before purchase.
Since most ratings and reviews are submitted by a wide array of users, maintaining them is hard and hence, we end up having a lot of incorrect/unwanted entries. Another issue we often face with simple methods of input is that users might unknowingly select the wrong option such as an incorrect rating or a genre for a video game. Reviews and descriptions don't face this issue since textual inputs have lesser tendency to be incorrectly entered, however not many people would be willing to spend their time adding textual information and hence we see a wide use of simple input methods. Recommendation systems are quite popular since they allow us to provide meaningful options to users for various purposes.

Deep learning has shown a lot of promise at this task. We define the G-Score of a game as a value that determines how good a game is which is derived from critic and user game ratings. In order to mitigate the issues mentioned earlier and to offer useful recommendations to users, we propose several deep neural network architectures that would predict the G-Score from the trailer and the summary of a video game. We believe that the use of summaries along with the trailers would aid the model to predict the G-Score better compared to the use of trailers alone. This would also aid game developers while creating trailers to see how well they score before a public release since the predicted G-Scores could be used to refine and improve the trailers. In order to train our models, we have created the VGD dataset and provide it along with this work.

### 3.2 Dataset

We have created a dataset named VGD that consists of the trailer, summary, developer, age rating, user-score, critic-score and genre of 1,950 video games. The data was collected from metacritic.com. The dataset along with the code used can be found at [https://goo.gl/Z8bNN3](https://goo.gl/Z8bNN3) for replicability and future use. This is the first dataset of its kind and we believe it would be quite helpful to the research community.

<table>
<thead>
<tr>
<th>Genre</th>
<th>Entries</th>
<th>Age Rating</th>
<th>Entries</th>
<th>G-Score (S)</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role-Playing</td>
<td>522</td>
<td>Mature</td>
<td>449</td>
<td>0-10</td>
<td>0</td>
</tr>
<tr>
<td>Strategy</td>
<td>329</td>
<td>Adults Only</td>
<td>2</td>
<td>11-20</td>
<td>3</td>
</tr>
<tr>
<td>Action</td>
<td>734</td>
<td>Everyone 10+</td>
<td>365</td>
<td>21-30</td>
<td>7</td>
</tr>
<tr>
<td>Sports</td>
<td>200</td>
<td>Teen</td>
<td>704</td>
<td>31-40</td>
<td>19</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>165</td>
<td>Everyone</td>
<td>409</td>
<td>41-50</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Other</td>
<td>522</td>
<td>51-60</td>
<td>263</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>61-70</td>
<td>433</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>71-80</td>
<td>693</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>81-90</td>
<td>416</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>91-100</td>
<td>27</td>
</tr>
</tbody>
</table>

Table 3.1 Distribution of VGD according to Genre classes, G-Score classes and Age Ratings
3.2.1 Preprocessing

The first step in preprocessing involved the removal of certain games that had missing details (a lot of games did not have trailers).

**Trailers** - The trailers extracted from the website had a resolution of 720p (with a few exceptions). We reduced the resolution to 360p since 720p required more space and mostly consisted of redundant information from the view of a neural network. We put an upper limit of 3 minutes for each trailer, trimming trailers that were larger to the 3-minute mark.

**Summaries** - We remove non-ASCII characters from the summaries since some of the summaries had terms from other languages like Japanese, Korean, French etc. However, since they are quite small in number, including them would not provide much value in terms of generalizability of the approach.

The final dataset consists of 1,950 video game trailers and summaries.

3.2.2 Statistics

Various statistics related to the dataset are provided, that show the diversity of the dataset. Video games were collected from a wide range of over 730 developers. Games span across various age ratings from E (Everyone) to M (Mature) which provides us a wide collection of games.

**Genres** - We cluster the genres into 5 groups based on similarity as specified in Table 3.2 and present the number of games belonging to each group in sub-tables in Table 3.1.

<table>
<thead>
<tr>
<th>Genre-Class</th>
<th>Genres</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role-Playing</td>
<td>Adventure, First-Person, Third-Person, Role-Playing</td>
</tr>
<tr>
<td>Strategy</td>
<td>Turn-Based, Strategy, War-Game, Puzzle, Platformer</td>
</tr>
<tr>
<td>Action</td>
<td>Action</td>
</tr>
<tr>
<td>Sports</td>
<td>Fighting, Sports, Racing, Wrestling</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>Simulation, Flight, Party, Real-Time</td>
</tr>
</tbody>
</table>

**Table 3.2** Our proposed grouping of genres into 5 classes based on similarity

**Game scores** - We define the G-Score of a game as an average of critic and user ratings, details are specified in Section 3.3. We observe that most games have G-Scores above 40 and only a small fraction of games have a G-Score below 40 as shown in Table 3.1. This results in some inter-class bias. The main reason for this is that most video games that would potentially have a bad G-Score would either not have trailers or not have any associated critic/user ratings since most people would not play the game in the first place and hence would not be present in the dataset.
Figure 3.1 An overview of our pipeline corresponding to Model-1. We start with the trailers and summaries as inputs and predict the G-Score classes as outputs. The Inception-V3 pre-trained model is used to extract features of video frames. ConvPoolBlocks and ConvBlocks are described in Figure 3.2. The output sizes at each of the layers are mentioned in the figure.

3.3 Score Prediction

Each video game has a user rating $R_u$ and critic rating $R_c$ associated with it. We define the G-Score of a game (S) as follows,

$$S = \frac{R_u + R_c}{2}$$

and aim to predict this G-Score. The G-Score essentially represents how good a game is.

Critic ratings are collected from a large number of critics and a weighted average is computed to form the final critic rating $R_c$. The weights of individual critics depend on the overall stature of the critic. Formally, if $R_{ci}$ corresponds to the rating of critic $i$ and $\alpha_i$ is the weight associated with critic $i$ and there are $M$ critics then,

$$R_c = \sum_{i=1}^{M} \alpha_i \cdot R_{ci}$$

The number of critics that review games vary from game to game since popular games often get a larger number of ratings as compared to others that are not so popular. Critic weights $\alpha_i$ are based on how well the critics performed in the past (well written, insightful reviews etc). This is determined by Metacritic staff who handle the website from where we collected our data.

The user rating $R_u$ is computed as an average of all user ratings submitted for the game. Formally, if $R_{ui}$ corresponds to the rating of user $i$ and there are $N$ users then,

$$R_u = \sum_{i=1}^{N} R_{ui}$$

Regardless of the number of users and critics that rate a game, we compute the final score as mentioned in Equation 3.1. We consider both trailers and summaries as inputs in order to predict this G-Score.
using our proposed model. We quantize the G-Scores to 10 classes since predicting the G-Score directly is a regression problem which is harder to tackle compared to classification problems.

### 3.3.1 Trailers

Each video game is associated with a trailer that we use in order to predict the G-Score.

**Trailer frame selection** - Since videos are captured with a frame-rate of 24 fps, it is infeasible to use them as they are since the sheer number of frames are too many. Hence, we propose a method to pick frames in a certain manner that would allow us to maximize the information we obtain from game trailers. Firstly, we reduce the frame-rate to 4 fps while extracting the frames from the video. We then follow the frame selection algorithm mentioned in Algorithm 1 in order to select frames. This allows us to capture important information at various parts of trailers. The reason we skip frames is that most trailers have a sequence of events that go on for a while before transitioning to the next sequence of events. Upon observation, we use a skip of 150 frames as a good approximation. We skip the first 50 frames since most trailers have textual information during the start of the video such as the developer titles, age ratings etc.

#### Algorithm 1 Frame selection for trailers:

1. Consider we have a set of $N$ frames $F_1, F_2, ..., F_N$
2. $f_{start} = 50$
3. while $f_{start} < N$
4. for $j = 0, j++, j < 10$
5. if $f_{start} + j < N$
6. Select frame $F_{f_{start}+j}$
7. else
8. Break
9. $f_{start}+ = 150$

**Trailer features** - We use the pre-trained Inception-V3 [59] model to extract features from each of the frames selected in the previous step. The model was pre-trained on ImageNet [13] and hence generalizes well to a wide range of images. We extract the features of the Avg.Pool layer (the penultimate layer in the network) which gives us a feature representation of 2048 elements per frame. Considering all the frames, we get a vector having dimensions $(M, 2048)$ where $M$ is the number of frames we selected by the frame selection algorithm as our final trailer features.

### 3.3.2 Summaries

Considering all the summaries we have, we create a dictionary where each word is given an index. We then go through each of the summaries replacing words with their corresponding indexes. Finally, we resize the summaries to a size of 100 by trimming the summaries if they are larger and padding them with zeros if they are smaller.
Figure 3.2 ConvPoolBlocks and ConvBlocks are used in order to process the summaries in our proposed models. A ConvPoolBlock consist of Convolution, Tanh, MaxPool and Dropout layers. A ConvBlock consists of Convolution, Tanh and Dropout layers.

3.3.3 Method

**Model-1** - We propose a deep learning based architecture as outlined in Figure 3.1 that uses a combination of recurrent and convolution based networks which allows us to process both trailer features and summaries in order to predict the G-Score of a video game. The frame features are fed to multiple levels of LSTMs\cite{21} that finally output a vector of size 512. The summaries are fed to an embedding layer that dynamically generates embeddings having a size of 300. These embeddings are then fed to a convolution-based network as depicted in Figures 3.1 and 3.2 that outputs a vector of size 512. Finally, these vectors are concatenated and passed along to a linear layer that outputs the G-Score class. We also perform experiments on multiple other model architectures, however, this gives us the best results.

**Model-2** - We use a time distributed CNN over extracted frame features to generate a small embedding for each frame which are then concatenated and fed to a fully connected layer in order to produce the final output vector. Summaries are passed through a CNN, similar to what was done earlier. The outputs from both the LSTM and CNN are concatenated and a linear layer is applied to predict the final output. One significant advantage of this approach was that the model had a very small number of parameters since the time distributed CNN shares weights across time. This would be an ideal model to use in memory constrained scenarios such as mobile computing.
Model-3 - We use a 3D CNN \cite{25} over the frames to generate an output embedding for the trailer. Summaries are passed through a CNN, similar to what was done earlier. The outputs are then concatenated and passed to a linear layer to predict the final G-Score class.

We also tried generating sentence embeddings using Doc2Vec \cite{32} for each of the summaries but they didn’t give us the best results and hence we stuck to dynamic embeddings as mentioned earlier. The three models mentioned here consider both the trailer and summary as inputs. In order to validate our claim that the use of summaries gives us accuracy improvements, we also perform the same experiments without considering summaries on each of the proposed models and report accuracies in Table \ref{table:5}. This shows that using summaries along with trailers gives us significant improvements of over 5%.

3.3.4 Implementation Details

We implemented the proposed models in the Keras \cite{10} framework over the Tensorflow \cite{11} backend. We use the Cross-entropy loss at the output of our model and use the Adam optimizer with a learning rate of 1e-4 and a decay of 1e-6 in order to train the model. We use tanh activations instead of ReLU throughout the model as it helps us achieve better accuracies. We also include multiple Dropout \cite{57} layers to allow the model to generalize well.

To evaluate our model, we perform 10-fold cross-validation and provide results. Further details on Model-1 (our best model) can be found in the code submitted along with this work at https://goo.gl/fYiElq.

3.4 Results

On each of our proposed models, we perform 10-fold cross-validation and consider the mean as our final accuracy. We observe a significant increase in accuracy with the inclusion of summaries as inputs along with the trailers. Model-1 gives us the best results in terms of accuracy. We believe the main reason for this is that Inception-V3 is trained on ImageNet which is a huge dataset of more than 1M images. Hence, it provides us with feature representations that are rich and meaningful.

Model-2 has a very small number of parameters which is why it is well suited for use in portable devices and memory constrained situations such as mobile processing. This, however, comes at a cost that the accuracy is lower than Model-1.

3.4.1 Qualitative Analysis

A few qualitative results have been provided where the network performs well in one case but fails at the other as shown in Figure \ref{figure:3}. Gran Turismo 6 has a true G-Score of 81 but we predict a G-Score class of 40-50. The main reason this fails is that the trailer has multiple overlay texts and clips which do not contain the game-play. A simple solution to frames containing overlay text is to ignore them before feeding them to model. We could also process these frames separately, extracting the text from them and using them as inputs along with the summaries of games. Since non game-play scenes
Figure 3.3 Qualitative examples where the model predicts the correct G-Score class for Super Mario Odyssey (left) and the incorrect G-Score class for Gran Turismo 6 (right). The true G-Score for Super Mario Odyssey is 93 and the true G-Score for Gran Turismo 6 is 81.

do not contribute any significant information when scoring a game, the model would misinterpret this information hence resulting in incorrect G-Scores. In the example provided, refer Figure 3.3, the frame containing the person would get a feature representation from Inception-V3 that has no relevance to the game and would ultimately contribute to noise. Handling non game-play scenes in trailers is an issue that is hard to tackle and is one of the shortcomings of this work. An approach towards this would be to train a model that takes a frame as an input and predicts if the frame is a game-play scene or not given the video as a reference.

3.4.2 Empirical Validation

We validate our claim that summaries provide information that is quite useful while predicting the G-Score of a video game. Hence, using both the trailers and summaries allows us to predict with a much better accuracy. We conduct a significance test where we perform experiments on predicting the G-Score of a video game based on the trailer alone and show that we gain significant accuracy improvements of over 5% when we use both the summaries and trailers in order to predict the G-Score as mentioned in Table 5.5. Most of the times, we have the summary at our disposal along with the trailers of games and hence, using information from multiple modalities helps us develop models that perform better.
<table>
<thead>
<tr>
<th>Model</th>
<th>Input</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-1</td>
<td>Trailer Only</td>
<td>65.2</td>
</tr>
<tr>
<td></td>
<td><strong>Trailer and Summary</strong></td>
<td><strong>70.5</strong></td>
</tr>
<tr>
<td></td>
<td>Improvement</td>
<td>+5.3%</td>
</tr>
<tr>
<td>Model-2</td>
<td>Trailer Only</td>
<td>63.3</td>
</tr>
<tr>
<td></td>
<td>Trailer and Summary</td>
<td>66.6</td>
</tr>
<tr>
<td></td>
<td>Improvement</td>
<td>+3.3%</td>
</tr>
<tr>
<td>Model-3</td>
<td>Trailer Only</td>
<td>64.5</td>
</tr>
<tr>
<td></td>
<td>Trailer and Summary</td>
<td>68.8</td>
</tr>
<tr>
<td></td>
<td>Improvement</td>
<td>+4.3%</td>
</tr>
</tbody>
</table>

Table 3.3 Results on predicting G-Scores using 10-fold cross-validation. We present mean accuracies on all three of our models considering trailers only as inputs as well as both trailers and summaries as inputs. Improvements obtained with each of the models are also mentioned.

3.5 Summary and Future Work

In this work, we show how valuable multimodal knowledge is at performing a task at hand. In most real-life scenarios we would have multimodal information available which could be utilized to train better models. We also provide a new VGD dataset that is a dataset on video games, a first of its kind. We propose multiple models that work under different scenarios such as memory constrained settings etc. We plan to apply our approach to movie trailers and summaries in order to show the generalizability of our approach. We plan to take care of overlay texts that occur in trailers by processing them separately in order to produce better results. Finally, we also plan to include audio in order to improve our prediction accuracies.
Chapter 4

Predicting the Genre and Rating of a Movie Based on its Synopsis

4.1 Background

Movies are one of the most prominent means of entertainment. The widespread use of the Internet in recent times has led to large volumes of data related to movies being generated and shared online. People often prefer to express their views online in English as compared to other local languages. This leaves us with a very little amount of data in languages apart from English to work on. To overcome this, we created the Multi-Language Movie Review Dataset (MLMRD). The dataset consists of genre, rating, and synopsis of a movie across multiple languages, namely Hindi, Telugu, Tamil, Malayalam, Korean, French, and Japanese. The genre of a movie can be identified by its synopsis. Though the rating of a movie may depend on multiple factors like the performance of actors, screenplay, direction etc but in most of the cases, synopsis plays a crucial role in the movie rating. In this work, we provide various model architectures that can be used to predict the genre and the rating of a movie across various languages present in our dataset based on the synopsis.

As the amount of data present online increases exponentially day by day, we have reached a point where a human cannot comprehend all of it in a meaningful manner due to its sheer size. This lead to work on automated recommender systems. The main issue with these kinds of methods is that not all the information is present online and all the information present need not be correct. Automated movie genre and rating prediction have a lot of applications. We can recommend same genre movies based on his previous watch history. Genre of a movie can be identified by its synopsis. Recommending a movie only based on its genre is not a good idea as the same genre can have both good and bad movies. So Recommending movies based on both genre and rating would result in a proper recommendation system. But the main problem here is that people do not often tend to rate the movie they watch, thus automated rating prediction would be of great help for recommendation systems. Though the rating of a movie depends on multiple factors like actors, screenplay, direction etc. but that information is very difficult to capture through available data. In most of the cases, synopsis of the movie plays a crucial impact on audience rating. In this paper, we propose multiple deep-learning based methods to predict the genre and rating of a movie based on its synopsis.
There is a very little amount of data in languages apart from English to work on. To overcome this, we created the Multi-Language Movie Review Dataset (MLMRD). The dataset consists of genre, rating, and synopsis of a movie across multiple languages, namely Hindi, Telugu, Tamil, Malayalam, Korean, French, and Japanese. Balance in the dataset is not that good because nowadays movies in specific languages tend to belong to only specific genres due to various reasons like movie collections, ease of making etc. For example, no documentary movies are present in Telugu as such movies make fewer collections at Tollywood box office.

<table>
<thead>
<tr>
<th>Class</th>
<th>Telugu</th>
<th>Hindi</th>
<th>Tamil</th>
<th>Malayalam</th>
<th>French</th>
<th>Japanese</th>
<th>Korean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>230</td>
<td>45</td>
<td>21</td>
<td>56</td>
<td>1,314</td>
<td>763</td>
<td>15</td>
</tr>
<tr>
<td>Comedy</td>
<td>60</td>
<td>35</td>
<td>27</td>
<td>25</td>
<td>2,602</td>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>Crime</td>
<td>8</td>
<td>10</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Drama</td>
<td>47</td>
<td>88</td>
<td>62</td>
<td>43</td>
<td>3,425</td>
<td>2,798</td>
<td>60</td>
</tr>
<tr>
<td>Family</td>
<td>18</td>
<td>0</td>
<td>0</td>
<td>21</td>
<td>0</td>
<td>763</td>
<td>0</td>
</tr>
<tr>
<td>Horror</td>
<td>20</td>
<td>0</td>
<td>17</td>
<td>9</td>
<td>208</td>
<td>278</td>
<td>0</td>
</tr>
<tr>
<td>Romance</td>
<td>133</td>
<td>42</td>
<td>19</td>
<td>14</td>
<td>127</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Thriller</td>
<td>43</td>
<td>33</td>
<td>0</td>
<td>18</td>
<td>532</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>Documentary</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>833</td>
<td>38</td>
<td>19</td>
</tr>
<tr>
<td>Rating</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>16</td>
<td>7</td>
<td>19</td>
<td>766</td>
<td>267</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>140</td>
<td>41</td>
<td>83</td>
<td>31</td>
<td>1,928</td>
<td>2,107</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>353</td>
<td>126</td>
<td>57</td>
<td>69</td>
<td>3,302</td>
<td>2,209</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>54</td>
<td>66</td>
<td>34</td>
<td>58</td>
<td>2,449</td>
<td>21</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>9</td>
<td>596</td>
<td>51</td>
<td>55</td>
</tr>
</tbody>
</table>

Table 4.1 Number of data-points per genre/rating for each language. Not all languages have data belonging to all classes, the classes not corresponding to a language are marked (in red) with zero entries.

4.2 Dataset

In order to create the dataset, we mined data from seven different websites. We used the data available in navbharattimes.indiatimes.com[1] for Hindi, telugu.samayam.com[2] for Telugu, tamil.samayam.com[3] for Tamil, malayalam.samayam.com[4] for Malayalam, m.movie.naver.com[5] for Korean, tsutaya.tsite.jp[6] for Japanese and www.allocine.fr[7] for French. We scraped the rating, genre, and synopsis of every movie from each website. Due to lack of resources and not much data is available in the specific language script, we could only mine a small amount of data. There aren’t a lot of regional sites available

that have trustworthy information to collect data from. Hence a lot of the languages have only a small number of data points in our dataset. However, we have ensured that the data collected, although small, is valid and collected from reputed movie review sites. We believe that having a small but strong and correct dataset is better than having a large dataset with a lot of noise and hence did not include other sites that did not have much reputation. The code can be found at https://goo.gl/nbWD9s and the dataset can be found at https://goo.gl/xpFv9q.

4.2.1 Data Extraction

The websites mentioned above have links to the synopsis of each movie along with the genre and rating in that web page. We first saved those links and then used beautiful soup to scrape the web page and get the synopsis, genre, and rating of the movie.

4.2.2 Preprocessing

After collecting all the required data, we had to pre-process the data to cluster genres into classes. Since the data collected had different classes in each language. We merged similar classes into one broader class. For example, the movies in the Biography class were moved into the Documentary class. Details of this grouping are provided in Table 4.2. Finally, we were left with 9 classes viz. action, comedy, crime, drama, family, horror, romance, thriller and documentary for each language. The details are mentioned in Table 4.1. We only added movies having all the three - genre, rating and synopsis into the dataset and ignored movies which were missing information. To validate the data, we performed a manual inspection at various data points selected at random to ensure the ratings and genres are valid and not erroneous.

4.2.2.1 Grouping of genres

The original data collected had several kinds of genres. We grouped all relevant genres together to finally end up with 9 different classes of genres as mentioned in Table 4.2.

4.2.3 Statistics

There are 14,991 entries in the dataset we compiled. The language-wise distribution is mentioned in Table 4.3. This is a big dataset covering a total of seven languages belonging to different language families.
<table>
<thead>
<tr>
<th>Genre Class</th>
<th>Genres</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Action, Adventure, Sci-Fi, Superhero, Sport, War</td>
</tr>
<tr>
<td>Comedy</td>
<td>Comedy, Romantic-Comedy</td>
</tr>
<tr>
<td>Crime</td>
<td>Crime</td>
</tr>
<tr>
<td>Drama</td>
<td>Drama, Fantasy, Music-Drama, Action-Drama</td>
</tr>
<tr>
<td>Family</td>
<td>Family, Animation, Musical, Anime, Kids</td>
</tr>
<tr>
<td>Horror</td>
<td>Horror</td>
</tr>
<tr>
<td>Romance</td>
<td>Romance, Music-Romance</td>
</tr>
<tr>
<td>Thriller</td>
<td>Thriller, Mystery</td>
</tr>
<tr>
<td>Documentary</td>
<td>Documentary, Autobiography, History, Biopic, Biography</td>
</tr>
</tbody>
</table>

Table 4.2 Preprocessing the genres to form 9 genre classes that are used for genre prediction.

<table>
<thead>
<tr>
<th>Language</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Telugu</td>
<td>559</td>
</tr>
<tr>
<td>Hindi</td>
<td>253</td>
</tr>
<tr>
<td>Tamil</td>
<td>181</td>
</tr>
<tr>
<td>Malayalam</td>
<td>186</td>
</tr>
<tr>
<td>French</td>
<td>9,041</td>
</tr>
<tr>
<td>Japanese</td>
<td>4,655</td>
</tr>
<tr>
<td>Korean</td>
<td>116</td>
</tr>
<tr>
<td>Total</td>
<td>14,991</td>
</tr>
</tbody>
</table>

Table 4.3 Size of the dataset created

4.3 Genre and Rating Prediction

We predict the genre and rating of a movie based on its synopsis alone. Genre prediction deals with 9 output classes as shown in Table 4.2. We treat rating prediction as a classification task rather than regression. We round the ratings leaving us with 5 classes that we try to predict.

4.3.1 Character Embeddings

Each character of the input synopsis is converted to a vector dynamically using an Embedding layer at the inputs to the networks. These character vectors are then passed along to various convolution and recurrent networks. Using a one-hot encoded representation of the characters also gave similar accuracies.

Convolution networks: The input to the CNN consists of all the character embeddings stacked as filters which are then passed along the network to predict an output genre/rating class. The network as
Figure 4.1 Branched CNN based genre/rating prediction model architecture with character inputs which considers the synopsis to perform the prediction. The Conv blocks represented in the figure consist of Convolution, ReLU and Max-pool layers. We used Dropouts at various places for regularization. Word inputs are similar except that the input size is different.

mentioned in Figure 4.1 has a branched structure where filters of various sizes are used in the convolution layers in each of the branches and the outputs are concatenated before being passed onto fully connected layers to predict the output genre/rating class.

Recurrent networks: For LSTM [21], GRU [11] and RNN [48] based networks as shown in Figure 4.2, we feed in character vectors one at a time as input and the predicted output is passed forward to multiple fully connected dense layers which predict the output genre/rating class.

4.3.2 Word Embeddings

Each word in the input is converted into a vector. These vectors are generated either dynamically using an Embedding layer or statically using Gensim [51]. These generated vectors are used as inputs to convolution and recurrent networks similar to how character encodings were used.

4.3.3 Sentence Embeddings

Sentence vectors were generated using Doc2Vec [33]. Doc2Vec takes all the sentences at once and generates sentence vectors for them. However, this requires all the data to be fed into Doc2Vec i.e both train and test sentences and hence this cannot be performed on unseen data.

Fully connected networks: Since the entire synopsis is encoded using a single vector, we pass the vector through a fully connected network which predicts the output genre/rating class and convolution/recurrent networks provide no benefits here.
Figure 4.2 LSTM based genre/rating prediction model architecture with character inputs which considers the synopsis to perform the predictions. Replacing LSTM cells with GRUs would give us GRU models.

4.3.4 Concatenated Embeddings

We observed that different types of embeddings performed well for different languages, for example, word embeddings for Telugu and Hindi, sentence embeddings for Tamil etc. Hence, concatenating all the three embeddings namely character, word and sentence embeddings and pass them through different models so that there can be an increase in the accuracy as the network chooses important parts of these embeddings.

4.4 Experiments

We performed numerous experiments using various deep learning models including convolution and recurrent based networks with character, word, and sentence level embeddings for inputs. We also compare our proposed models with some of the popular traditional approaches such as SVMs [12] and Random Forests [58] and show that deep learning based methods beat them by large margins as shown in Tables 4.4 and 4.5.

4.4.1 Experimental Details

We use Keras with the Tensorflow backend to perform all our experiments. We use a GeForce GTX-1080Ti GPU in order to train our models (Each model takes less than 15 minutes to complete training). We use dropouts at various locations in the networks to reduce over-fitting. Categorical cross entropy loss is used as the loss function along with the Adam optimizer for training all the networks. We observe that dynamic embeddings perform better than static embeddings in all word based models and hence we use embedding layers in all the models instead of using Gensim or GloVe word vectors. ReLU activations are used throughout the networks except for the last layers which use SoftMax activations in all the models. The code provided along with the paper has further implementation details.
4.4.1 SVM

SVMs are commonly used for recognition and regression analysis of data. Considering features from the reviews as inputs, they try to classify them into one of the genre and rating classes. We run a trained Doc2Vec model using the entire review as an input and that provides us a 300-dimensional embedding that we use as an input to the SVM.

4.4.1.2 Random Forests

Similar to the features we use for the SVM, we use the Doc2Vec embeddings of reviews as inputs for the Random Forest classifier that predicts the genre and rating classes.

4.4.1.3 CNN

We use dropouts of 0.5, 0.7, and 0.8 at three places in the network to ensure that the model does not overfit.

Character embedding based model: Inputs are padded to a length of 300 characters and trimmed if they exceed this length. Character embeddings are generated using an embedding layer that generates 300-dimensional embeddings for each character. We train the model for 300 epochs with a batch size of 512.

Word embedding based model: Inputs are padded to a length of 150 words and trimmed if they exceed this length. Word embeddings are generated using an embedding layer that generates 300-dimensional embeddings for each word. We train the model for 200 epochs with a batch size of 512.

4.4.1.4 LSTM

Character-based inputs are padded to a length of 300 and word-based inputs are padded to a length of 100. The embedding layer generates 300-dimensional embeddings. The network consists of two LSTM layers followed by multiple Dense layers. A recurrent dropout of 0.4 is used in the first LSTM layer. We train the models for 300 epochs with a batch size of 512.

4.4.1.5 GRU

The parameters used are identical to the LSTM parameters except that both the LSTM layers are replaced with GRU layers.

4.4.1.6 FCNN

The model receives a single 300-dimensional sentence embedding that was generated using Doc2Vec. This is passed through a few Dense layers to get our final output. We use dropouts of 0.4 at various places in the network. We train the models for 200 epochs with a batch size of 512.
Figure 4.3 Hybrid model based on stacking three models to predict genre/rating with word, character and sentence embeddings as the input considers synopsis to perform the prediction. We used Dropouts at various places for regularization.

4.4.1.7 Hybrid Model

The model as shown in Figure 4.3 receives three inputs i.e. word embedding, char embedding and sentence embedding. The word and char embeddings go into two different LSTM networks. The sentence embeddings go into a fully connected dense network. Each model produces a 100-dimensional output which are concatenated. This 300-dimensional concatenated embedding is given as an input to a dense network to predict the genre and ratings. We use a Dropout of 0.4 throughout the model and train the model for 300 epochs with a batch size of 512.

<table>
<thead>
<tr>
<th>Model</th>
<th>Train %</th>
<th>Telugu</th>
<th>Hindi</th>
<th>Tamil</th>
<th>Malayalam</th>
<th>French</th>
<th>Japanese</th>
<th>Korean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Char-CNN</td>
<td>87.7</td>
<td>87.5</td>
<td>89.2</td>
<td>87.8</td>
<td>89.0</td>
<td>93.3</td>
<td>89.8</td>
<td></td>
</tr>
<tr>
<td>Word-CNN</td>
<td>88.9</td>
<td>87.9</td>
<td>87.6</td>
<td>88.3</td>
<td>89.0</td>
<td>91.0</td>
<td>90.7</td>
<td></td>
</tr>
<tr>
<td>Concat-CNN</td>
<td>89.7</td>
<td>89.3</td>
<td>89.5</td>
<td>88.9</td>
<td><strong>89.4</strong></td>
<td>92.4</td>
<td>89.8</td>
<td></td>
</tr>
<tr>
<td>Char-LSTM</td>
<td>87.7</td>
<td>88.0</td>
<td>87.9</td>
<td>88.1</td>
<td>89.1</td>
<td><strong>93.8</strong></td>
<td>91.7</td>
<td></td>
</tr>
<tr>
<td>Word-LSTM</td>
<td>89.1</td>
<td>88.7</td>
<td>87.8</td>
<td>88.2</td>
<td>88.9</td>
<td>91.6</td>
<td>91.2</td>
<td></td>
</tr>
<tr>
<td>Concat-LSTM</td>
<td>89.1</td>
<td>88.8</td>
<td>89.8</td>
<td>89.7</td>
<td>88.8</td>
<td>92.4</td>
<td>90.7</td>
<td></td>
</tr>
<tr>
<td>Char-GRU</td>
<td>87.4</td>
<td>87.5</td>
<td>87.4</td>
<td>87.6</td>
<td>88.9</td>
<td>93.5</td>
<td>91.7</td>
<td></td>
</tr>
<tr>
<td>Word-GRU</td>
<td>87.3</td>
<td>87.8</td>
<td>87.6</td>
<td>87.3</td>
<td>89.0</td>
<td>90.8</td>
<td>90.7</td>
<td></td>
</tr>
<tr>
<td>Concat-GRU</td>
<td>88.9</td>
<td>88.9</td>
<td>89.5</td>
<td>89.5</td>
<td>88.9</td>
<td>92.0</td>
<td>88.9</td>
<td></td>
</tr>
<tr>
<td>Sent-FCNN</td>
<td>87.6</td>
<td>88.1</td>
<td>89.3</td>
<td>87.0</td>
<td>89.4</td>
<td>92.8</td>
<td><strong>92.6</strong></td>
<td></td>
</tr>
<tr>
<td>Concat-FCNN</td>
<td>83.9</td>
<td>84.3</td>
<td>89.8</td>
<td>82.5</td>
<td>84.3</td>
<td>92.1</td>
<td>85.0</td>
<td></td>
</tr>
<tr>
<td>Hybrid-model</td>
<td><strong>91.2</strong></td>
<td><strong>89.9</strong></td>
<td><strong>90.0</strong></td>
<td><strong>89.8</strong></td>
<td><strong>89.8</strong></td>
<td>91.8</td>
<td>92.1</td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>41.9</td>
<td>45.1</td>
<td>48.6</td>
<td>39.4</td>
<td>49.0</td>
<td>63.2</td>
<td>62.5</td>
<td></td>
</tr>
<tr>
<td>Random Forests</td>
<td>53.5</td>
<td>50.9</td>
<td>40.5</td>
<td>42.1</td>
<td>40.5</td>
<td>67</td>
<td>58.3</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4 Genre prediction accuracies for various models on each of the languages in MLMRD.
4.4.2 Results

Promising results were obtained in both genre and rating prediction using just the synopsis as the input, as presented in Tables 4.4 and 4.5. For instance, we obtain 91.2% and 90.2% while predicting the Genre and Rating in Telugu respectively.

4.4.3 Analysis

[14] classified languages in 6 ways depending on whether a subject follows a verb or whether an object follows a verb. The languages we worked on come under SV/OV(Subject-Object-Verb) type of languages. Our dataset consists of multiple languages, some of which are agglutinative (Telugu, Malayalam, Tamil, Japanese and Korean). Our methods obtain good results with various types of languages.

Character vs Word Embeddings: We observe that on the whole, word embeddings perform better in general, however in certain cases considering agglutinative languages such as genre prediction in Japanese and rating prediction in Malayalam perform better with character embeddings.

Sentence FCNNs: Datasets having small amounts of data work well with sentence vectors. Larger datasets, however, pose issues since the embeddings generated are not precise enough in these cases to differentiate the inputs well.

Hybrid Model: LSTM networks learn sequence-based information very well from the character and word embeddings whereas the FCNN learns well from the sentence vectors. Our intuition was that if we develop a model which would use these three models collectively to predict the genre and rating there would be a significant increase in accuracy. So we developed a stacked model which uses combined information from two LSTM networks and one FCNN network to predict the genre and rating. Stacking is an ensemble learning technique and is also known as meta ensembling. This new model outperforms the earlier models as it gives more weight to the individual model where it performs well and gives lesser weight to the individual model where it performs badly. The reason we cannot see a huge change

Table 4.5 Rating prediction accuracies for various models on each of the languages in MLMRD.

<table>
<thead>
<tr>
<th>Model</th>
<th>Train %</th>
<th>Telugu</th>
<th>Hindi</th>
<th>Tamil</th>
<th>Malayalam</th>
<th>French</th>
<th>Japanese</th>
<th>Korean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Char-CNN</td>
<td></td>
<td>85.2</td>
<td>83.1</td>
<td>82.7</td>
<td>82.1</td>
<td>89.1</td>
<td>90.5</td>
<td>90.4</td>
</tr>
<tr>
<td>Word-CNN</td>
<td>82.2</td>
<td></td>
<td>80.1</td>
<td>83.2</td>
<td>81.1</td>
<td>89.0</td>
<td>89.4</td>
<td>89.8</td>
</tr>
<tr>
<td>Concat-CNN</td>
<td>90.2</td>
<td></td>
<td>89.5</td>
<td>89.2</td>
<td>89.5</td>
<td>89.2</td>
<td>91.5</td>
<td>89.8</td>
</tr>
<tr>
<td>Char-LSTM</td>
<td>85.9</td>
<td></td>
<td>81.7</td>
<td>82.2</td>
<td>82.1</td>
<td>89.1</td>
<td>89.1</td>
<td>93.5</td>
</tr>
<tr>
<td>Word-LSTM</td>
<td>86.2</td>
<td></td>
<td>83.5</td>
<td>80.4</td>
<td>83.2</td>
<td>88.9</td>
<td>89.0</td>
<td>91.4</td>
</tr>
<tr>
<td>Concat-LSTM</td>
<td>89.0</td>
<td></td>
<td>88.9</td>
<td>88.9</td>
<td>89.2</td>
<td>88.9</td>
<td>91.2</td>
<td>93.5</td>
</tr>
<tr>
<td>Char-GRU</td>
<td>85.8</td>
<td></td>
<td>82.4</td>
<td>81.2</td>
<td>81.1</td>
<td>89.2</td>
<td>89.5</td>
<td>91.2</td>
</tr>
<tr>
<td>Word-GRU</td>
<td>86.1</td>
<td></td>
<td>83.2</td>
<td>81.1</td>
<td>80.5</td>
<td>89.0</td>
<td>89.1</td>
<td>91.7</td>
</tr>
<tr>
<td>Concat-GRU</td>
<td>88.9</td>
<td></td>
<td>89.5</td>
<td>88.9</td>
<td>89.5</td>
<td>88.9</td>
<td>91.2</td>
<td>90.7</td>
</tr>
<tr>
<td>Sent-FCNN</td>
<td>85.2</td>
<td></td>
<td>80.9</td>
<td>84.2</td>
<td>80.2</td>
<td>89.2</td>
<td>89.5</td>
<td>91.3</td>
</tr>
<tr>
<td>Concat-FCNN</td>
<td>80.0</td>
<td></td>
<td>78.0</td>
<td>89.1</td>
<td>69.5</td>
<td>73.1</td>
<td>77.9</td>
<td>73.3</td>
</tr>
<tr>
<td>Hybrid-model</td>
<td>84.6</td>
<td></td>
<td>83.1</td>
<td>81.0</td>
<td>80.0</td>
<td>80.5</td>
<td>80.8</td>
<td>82.5</td>
</tr>
<tr>
<td>SVM</td>
<td>80.0</td>
<td>50.9</td>
<td>45.9</td>
<td>47.3</td>
<td></td>
<td>44.7</td>
<td>48.8</td>
<td>45.8</td>
</tr>
<tr>
<td>Random Forests</td>
<td>68.8</td>
<td>52.9</td>
<td>54.0</td>
<td>44.8</td>
<td>43.1</td>
<td>55.5</td>
<td>54.0</td>
<td></td>
</tr>
</tbody>
</table>

27
in the rating prediction unlike genre prediction is that the final objective function is not able to learn well from three different models due to the difference in the flow of gradients. If the training is done separately, we would see an increase in accuracy.

**Traditional vs Deep Learning approaches:** There is a huge difference in the performance of traditional machine learning approaches such as SVMs and Random Forests when compared to deep learning based methods such as Sent-FCNN. They all use the same inputs which are the embeddings generated using the Doc2Vec model. We believe that one of the main reasons for this is that deep learning based approaches tend to generalize a lot better as compared to traditional methods and hence they perform a lot better on unseen test data. We have also tried to see how these methods perform if the testing data is just a subset of training data. In this case, we notice that they are both able to represent their training data well and hence achieve similar accuracies during testing. However, this only happens since the number of data points in our dataset are not a lot. When the number of data points increases, deep learning based approaches show tremendous amounts of generalizability which allows them to attain much higher accuracies compared to traditional methods. We validate this by adding noise to our inputs by randomly scaling the inputs up to 15%. This resulted in the failure of traditional approaches as we discussed earlier.

**Inter-language comparisons:** Having a dataset that consists of multiple languages allows us to verify how well our approaches scale and how they can be generalized and applied to data from various domains. MLMRD would also be useful to other researchers who would want to test out their approaches on different languages since multilingual data is becoming popular in recent times. We observe that Telugu, French, Japanese, and Korean perform much better than Hindi, Tamil, and Malayalam in rating prediction. This dataset would also allow us to work towards generalized methods that work on multiple forms of inputs that don’t require different models to handle different languages which is how traditional approaches work.

### 4.5 Summary

We provide the multi-lingual dataset MLMRD, consisting of movie genres, ratings and the synopsis which can be used to test various machine learning and NLP based techniques on different kinds of data. We believe that this would be a valuable asset since a lot of these languages are low-resource languages with almost no data available to experiment on. We also propose multiple methods to establish baselines for movie genre and rating prediction based on the synopsis. Additionally, we show how our proposed methods are generalizable and work well on different kinds of data. We plan to extend the approach using movie plots as inputs which would provide us with important information. We also plan to normalize the data collected so that each of the classes have a similar number of data-points.
Chapter 5

Sentiment as a prior for movie rating prediction

5.1 Background

Movie ratings play an important role in tasks such as user movie recommendations, verifying the relationship between user-submitted reviews and ratings etc. The ability to predict the rating of a movie would be useful considering these aspects. In this work, firstly, we propose methods to predict the movie rating based on its summary. We then set out to use priors that are generally available with movie summaries in order to improve the accuracy. In order to achieve this, we consider the associated movie reviews as well while predicting the rating and provide insights on why this helps our models perform better. We use the review based sentiment along with the summary in order to predict the rating more accurately since the sentiment captures a lot of essential information that can aid rating prediction. We experiment with various deep learning architectures and the results show a significant accuracy boost of around 2% in most of the models which show the generalizability of our approach.

The widespread usage of the internet has enabled people to share their views with the rest of the world online. This method of broadcasting opinions has gained a lot of popularity ever since. However, this led to a decrease in the quality of opinions that were shared. Due to this, people find it challenging and difficult to browse through all the opinions. This issue of bogus and random opinions is witnessed in a lot of cases where the user can provide feedback in a quick manner such as multiple choice options, checkboxes etc. Movie ratings fall in this category. Opinions requiring a text description etc such as movie reviews, however, would be much less prone to incorrect/invalid responses. Predicting the rating of a movie using the reviews along with the summary would help provide and counter these invalid ratings which cause issues in a lot of applications such as personalized movie recommendation engines etc. We propose a solution to this problem where we predict a rating for a movie considering the features embedded in the reviews along with the summaries. The sentiment of reviews is a valuable feature that we use in order to solve the task at hand better. This would lead to more accurate rating predictions and would help people make the right decisions. Traditional methods such as SVMs and logistic regression do not perform well at the given task and hence we propose a deep learning based method. We verified and validated our method on Telugu movie reviews. Telugu is an agglutinative language for which very
few resources and tools are available. It is morphologically very rich which makes it hard to perform natural language processing related tasks on it.

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Pronouns</th>
<th>Conjunctions</th>
<th>Nouns</th>
<th>Verbs</th>
<th>Adjectives</th>
<th>Adverbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reviews - All</td>
<td>3,900</td>
<td>1,174</td>
<td>62,618</td>
<td>25,803</td>
<td>3,558</td>
<td>7,603</td>
</tr>
<tr>
<td>Reviews - Unique</td>
<td>355</td>
<td>23</td>
<td>8,946</td>
<td>5,740</td>
<td>250</td>
<td>769</td>
</tr>
<tr>
<td>Summaries - All</td>
<td>984</td>
<td>476</td>
<td>20,271</td>
<td>6,944</td>
<td>1,310</td>
<td>1,697</td>
</tr>
<tr>
<td>Summaries - Unique</td>
<td>180</td>
<td>15</td>
<td>3,811</td>
<td>2,377</td>
<td>146</td>
<td>308</td>
</tr>
<tr>
<td>Both - All</td>
<td>4,884</td>
<td>1,650</td>
<td>82,889</td>
<td>32,747</td>
<td>4,868</td>
<td>9,300</td>
</tr>
<tr>
<td>Both - Unique</td>
<td>395</td>
<td>24</td>
<td>10,183</td>
<td>6904</td>
<td>288</td>
<td>902</td>
</tr>
</tbody>
</table>

Table 5.1 Dataset statistics for both the sentiment classification and the rating prediction datasets created. Sentiment classification dataset consists of reviews whereas rating prediction dataset consists of reviews and summaries together.

5.2 Dataset

Telugu, being a low resource language, we had to create two datasets for training our models. We used 123telugu.com\(^1\) to collect the data required. The website presents data in a specific format as follows - Story, Positive Points, Negative Points, Technical Section, Conclusion and Summary. We created two separate datasets. The first is the Telugu movie rating dataset consisting of summaries, sets of reviews and ratings and the other is the Telugu sentiment classification dataset which consists of positive and negative reviews. The latter is used to train the sentiment classifier and is used as a prior for rating prediction. The dataset along with the source code can be found at https://goo.gl/WBWdT8.

5.2.1 Analysis of the Dataset

Telugu movie rating dataset consists of multiple movie summaries, their corresponding ratings, and reviews spanning across 473 movies. There are a total of 18,502 unique words in this dataset.

<table>
<thead>
<tr>
<th>Rating</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>127</td>
</tr>
<tr>
<td>3</td>
<td>303</td>
</tr>
<tr>
<td>4</td>
<td>33</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 5.2 Count of movies per rating class present in the dataset

Telugu sentiment classification dataset consists of annotated positive and negative reviews. It has 5,684 positive reviews and 4,553 negative reviews. There are a total of 16,001 unique words in this dataset.

\(^1\)http://www.123telugu.com/reviews/main/more_reviews.html
5.2.2 Dataset statistics

We analyzed the dataset and collected various statistics corresponding to the number of nouns, verbs etc in Table 5.1 present in the dataset. We also analyze the year-wise span of movie release dates in Table 5.3 and the number of movies corresponding to each of the rating classes 1-5 in Table 5.2. This shows the richness of the dataset and that it spans out over time which shows better diversity. This is the first dataset of its kind in Telugu. We used a parts of speech tagger developed by Siva Reddy [50] to generate some of the statistics.

<table>
<thead>
<tr>
<th>Year</th>
<th>2012</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
<th>2016</th>
<th>2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>28</td>
<td>93</td>
<td>84</td>
<td>113</td>
<td>150</td>
<td>48</td>
</tr>
</tbody>
</table>

Table 5.3 Time-span of movies released per year present in the dataset

5.3 Sentiment prediction

Sentiment prediction has been a great area of research in the recent times and is a challenging task especially in morphologically rich languages. The task requires us to classify a given sentence either as “Positive” or ”Negative”. In order to do this, we went ahead and tried out multiple deep learning based methods, however, we got the best results with a word-level multi-layer Convolutional Neural Network(CNN) which we used as our final model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Acc(50-50)</th>
<th>Acc(80-20)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCNN</td>
<td>85.54</td>
<td>87.30</td>
</tr>
<tr>
<td>CNN</td>
<td><strong>91.05</strong></td>
<td><strong>92.47</strong></td>
</tr>
<tr>
<td>RNN</td>
<td>89.02</td>
<td>90.19</td>
</tr>
<tr>
<td>LSTM</td>
<td>90.59</td>
<td>91.64</td>
</tr>
</tbody>
</table>

Table 5.4 Sentiment classification experiments with various models on 50-50 and 80-20 train-test data splits.

5.3.1 Recurrent networks

We implemented single-layer and multi-layer Recurrent Neural Networks(RNNs) [48], Long Short Term Memory cells(LSTMs) [21] and Gated Recurrent Units(GRUs) [11] where the inputs were processed into character embeddings and word embeddings before passing them to the networks. Character embeddings were simple one hot encoded vectors whereas word embeddings were dynamically generated using embedding layers in our networks and also using static embeddings generated by Word2Vec (Gensim) [51]. We got the best results with dynamic word based embeddings. However, convolution networks beat them, refer to Table 5.4.
Figure 5.1 Branched CNN based rating prediction model architecture which considers sentiment priors along with the summaries to predict the ratings. The Conv blocks represented in the figure consist of Convolution, ReLU and Max-pool layers. We used Dropouts at various places for regularization.

5.3.2 Convolution networks

Similar to recurrent networks we processed the inputs into character and word embeddings and stacked them across as filters before passing them to the network. Dynamic word based embeddings gave us the best results and they even beat LSTM dynamic word based embeddings which were the best in recurrent networks for the task. Hence our final trained model used for sentiment prediction consisted of a convolution network with multiple convolution, dropout, relu, max-pool and fully connected layers.

5.4 Rating prediction

We could predict the rating of a given movie just based on its summary but quite often, we have reviews for the movie along with their summary which could be utilized in order to improve the prediction capability of our networks. Using the sentiment of these reviews as a prior along with the summary aids the task at hand. In order to generate sentiment priors, we used the sentiment classification network mentioned in the previous section.

The actual task of predicting the rating is a regression problem where the output would be a single floating value between 0.0 to 5.0, in order to simplify the task, we convert it into a classification problem where we round the true rating to the nearest integer which would give us a five-class classification problem.

5.4.1 Prediction without priors

Prediction of the rating class using just the summaries alone without any other information about the movie.
5.4.1.1 Character embeddings

Each character of the input summary is converted to a vector dynamically using an Embedding layer at the inputs to the networks. These character vectors are then passed along to various convolution and recurrent networks. Using a one-hot encoded representation of the characters also gave similar accuracies.

Convolution networks: The input to the CNN consists of all the character embeddings stacked as filters which are then passed along the network to predict an output rating class. The network has a branched structure where filters of various sizes are used in the convolution layers in each of the branches and the outputs are concatenated before being passed onto fully connected layers to predict the output rating class.

![Figure 5.2 LSTM based rating prediction model architecture which considers sentiment priors along with the summaries to predict movie ratings](image)

Recurrent networks: For LSTM, GRU and RNN based networks we feed in character vectors one at a time as input and the predicted output is passed forward to multiple fully connected dense layers which predict the rating class.

5.4.1.2 Word embeddings

Recurrent networks: For LSTM, GRU and RNN based networks we feed in character vectors one at a time as input and the predicted output is passed forward to multiple fully connected dense layers which predict the rating class.

5.4.1.3 Sentence embeddings

Sentence vectors were generated using Doc2Vec. Doc2Vec takes all the sentences at once and generates sentence vectors using an unsupervised algorithm. However, the generation of sentence vectors requires all the data to be fed into Doc2Vec i.e both train and test sentences and hence this cannot be
performed on unseen data since it’s not feasible to generate vectors on the fly without having the entire dataset. Hence the test data used while classification would not be considered as truly unseen data.

**Fully connected networks** Since the entire summary is encoded using a single vector, there is no advantage in using convolution or recurrent based networks here. Hence, we just pass the sentence vector through a few fully connected networks which finally predict the rating class.

### 5.4.2 Prediction with sentiment priors

Each summary is also associated with a set of comments or one line reviews that complement the summary and hence can help us predict the rating better. For each of these reviews, we compute the sentiment using our sentiment prediction network that we have described in section 4. The network gives us a probability of the reviews being a positive review. We create a vector of these probabilities for all the reviews present. This vector acts as our sentiment prior which we use to predict the rating. We have also tried getting an average of the sentiment for all the reviews and using that as a prior instead but that did not give us the best results since we lose a lot of information in this process about the individual reviews themselves. Since all the movie summaries don’t have the same number of reviews associated with them, we pad the priors to a certain length with a neutral sentiment while creating the sentiment priors.

#### 5.4.2.1 Inclusion of priors

Now that we have the priors defined, we need to decide where to include them in the networks. Considering the convolution networks, inserting the priors before the convolution layers doesn’t work well since it’s not essentially a filter associated with the input summary. In recurrent networks, passing on the priors (after a re-projection into the required shape using a dense layer) and appending them to the input did not give them enough weight and they were not able to influence the classification much. Finally, we decided to append the sentiment priors before the fully connected layers of both convolution (refer to Figure 5.1) and recurrent networks (refer to Figure 5.2). Concatenating the sentiment features gave us the best results, however, we also tried other approaches such as Multiplying and Adding the projected sentiment features with the fully connected layer outputs of various networks after passing the summary along etc, but concatenation ensured that none of the features are lost and they gave us the best results.

### 5.4.3 Empirical validation and explanation

To validate our claim that sentiment priors act as one of the most important features in the prediction of a movie rating, we train simple fully connected networks to predict the rating of a movie using just the sentiments from reviews corresponding to the movie. These results confirm our claim and hence
we proceed to use the sentiment of reviews as priors along with the summaries to predict the ratings of movies.

5.5 Experiments

We performed various experiments on the datasets created and reported the accuracies achieved in Table 5.5. We also tried to pass sub-word level embeddings as inputs to an LSTM which were generated using a CNN but did not achieve any significant improvements in the results mainly because Telugu, unlike English, is an agglutinative language.

We use Keras [9] on top of the Tensorflow backend in order to implement all the models mentioned. We use a categorical cross-entropy loss and the Adam optimizer with a learning rate of 0.001 for all our models. We use a batch size of 512 for all models. RNN based models and character embedding based models are trained for 1000 epochs whereas the rest of the models are trained for 200 epochs.

<table>
<thead>
<tr>
<th>Model</th>
<th>No Prior</th>
<th>With Prior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>65.64</td>
<td>-</td>
</tr>
<tr>
<td>Random Forests</td>
<td>67.90</td>
<td>-</td>
</tr>
<tr>
<td>Polynomial SVM</td>
<td>63.15</td>
<td>-</td>
</tr>
<tr>
<td>Word-CNN</td>
<td>89.47</td>
<td>90.74</td>
</tr>
<tr>
<td><strong>Word-LSTM</strong></td>
<td>90.32</td>
<td><strong>92.40</strong></td>
</tr>
<tr>
<td>Word-GRU</td>
<td>89.68</td>
<td>91.32</td>
</tr>
<tr>
<td>Char-CNN</td>
<td>84.21</td>
<td>85.89</td>
</tr>
<tr>
<td>Char-LSTM</td>
<td>85.62</td>
<td>85.68</td>
</tr>
<tr>
<td>Char-GRU</td>
<td>85.68</td>
<td>85.99</td>
</tr>
<tr>
<td>Sent-FCNN</td>
<td>90.48</td>
<td>92.38</td>
</tr>
</tbody>
</table>

Table 5.5 Results on rating prediction for movies. Considering an 80-20 train-test split

5.5.1 Results

We notice that traditional methods are outperformed significantly by deep learning based methods. We also see a significant improvement in accuracies considering the priors as well while predicting the ratings in most models and nominal accuracy improvements in the remaining models.

5.5.1.1 Analysis

From the results obtained, we see that word-level models gave better accuracies than character-level models in general. Character-level model consider one character at a time, hence having a large input size compared to word level models. Recurrent networks face issues with sequences that are too long and hence don’t perform too well with character-level encodings. Convolution networks do not ensure temporal relations since there is no concept of memory involved there and hence having character level
input would imply that features generated by the embedding layer (or one hot vectors) are not the best possible representations of the input data and hence perform poorly. Word-embeddings however, prove to work much better in both convolution and recurrent networks.

Sentence embeddings proved to be of a great advantage when used with fully connected neural networks but as the size of the dataset increases, these vectors become less and less useful since the entire sentence (of varying size) is forced to be described in terms of a fixed sized embedding which is a limitation of these embeddings.

We observe that LSTMs and GRUs outperform RNNs due to various limitations such as vanishing or exploding gradients etc that have been discussed in various papers such as [21] in the past.

On the whole, we see that considering priors always improve the accuracies in all the models as stated in Table 5.5. This confirms our claim that review sentiment is an important prior for rating prediction of movies.

5.6 Summary

Considering the sentiment of the reviews as a prior increased the accuracy by around 2% in various models. We managed to achieve good results even though Telugu is an agglutinative language which makes the task a challenging one. We plan to increase the priors i.e taking into consideration the actors, genre, directors, plot of the movie etc. which would help us boost the accuracy further. We also plan to extend this to other languages similar to Telugu such as Turkish, Korean, Japanese, Mandarin and other Dravidian languages.
6.1 Background

We have progressed to a time where the things we say and the way in which we express ourselves is of utmost importance, especially with emerging technologies related to speech synthesis, etc. Most often than not, it’s not just the meaning of something we say that matters but the way in which we put it (including the sentiment and emotion) is of larger value. We can find multiple examples of such scenarios in our day to day lives. Consider the case when we talk to elders or someone senior in a profession, we would like to address them formally and not want to accidentally refer to them in the same way we might with friends. Another scenario could be a meeting where members from different countries meet up and would like to talk to each other with the help of human translators, this works well since the human is capable of understanding the nuances of each language and hence can help translate correctly and efficiently. Machine translators, however, face a lot of issues in these areas. They are unable to capture the sentiment behind the sentence and hence we often end up with cases where the meaning is correct but the tone of the sentence or its emotion is not the same as the source sentence. We feel this is an important aspect that one needs to work on and hence have come up with an amended definition of the BLEU score which we call the SEMMT score that takes into consideration how appropriately the sentiment is transferred as well. Using this score as a metric for machine translators, we would be able to train and model translators that are closer to human translators which can help us ensure that the sentiment is transferred correctly.

6.2 Popular metrics

There are multiple metrics that help us evaluate the quality of a translation. Some of the popular ones are BLEU, CIDEr, ROUGE and METEOR. Each of these metrics have their own advantages and disadvantages.
6.2.1 BLEU

Bilingual evaluation understudy (BLEU) is one of the most popular methods to evaluate translation methods. The score helps us evaluate how well our translation methods perform by taking a candidate sentence that is the output of our translation method on some input text and then comparing it with one or multiple reference texts which are those that we deem are correct as mentioned in [46]. Simply put, n-grams from each candidate and reference are computed and then we find the number of matches. The more the matches, the better the candidate is and hence gets a higher score.

6.2.2 CIDEr

[62] talks about Consensus-based Image Description Evaluation (CIDEr). A score for n-grams of length n is computed using the average cosine similarity between the candidate sentence and the reference sentences, which accounts for both precision and recall.

6.2.3 METEOR

Metric for evaluation of translation with explicit ordering (METEOR) evaluates a translation by computing a score based on explicit word-to-word matches between the translation and a reference translation as explained in [6]. If more than one reference translation is available, the given translation is scored against each reference independently, and the best score is reported.

6.2.4 ROUGE

[39] speaks about Recall-oriented understudy for gisting evaluation (ROUGE), a metric which determines the quality of summaries/translations by comparing it to other (ideal) summaries/translations created by humans. The measures count the number of overlapping units such as n-gram, word sequences, and word pairs between the computer-generated summary to be evaluated and the ideal summaries created by humans.

6.3 Dataset

For all testing and evaluation purposes, we used the Opensubtitles2018[1] dataset as our golden standard reference. This is a large dataset which provides us with German to English correspondences that could be used to evaluate the SEMMT metric that we designed. The dataset has a total of 22,512,639 entries.

We performed some preprocessing on the data such as converting all the text to lower case and removing the punctuation. We also removed any unknown symbols/characters by replacing them with white-space hence effectively ignoring them in order to maintain uniformity throughout.

We extracted 3,000 subtitle pairs from the dataset and segregated them into three classes viz. positive, negative and neutral. Section 6.4.1 speaks about the tools used to calculate the sentiment of subtitles. The positive class consists of subtitles which are positive in both German and English i.e we took the subtitles whose sentiment is retained even after translation. It was made sure that each class had exactly 1,000 subtitle pairs. We had these subtitles manually annotated in order to check if our formulation takes human judgment into consideration. Details of how the manual annotation was performed etc are explained in Section 6.6.1.

6.4 Sentiment

Sentiment analysis has mainly been performed on English texts in the past. This lead to an abundance of sentiment resources in English but minimal to no resources in other languages. Some of the common approaches to performing sentiment analysis in other languages often includes translating the text to English (or another resource-rich language) and then perform sentiment analysis. This further shows why preserving sentiment is really necessary for translation systems. [61] shows an increase in accuracy of rating prediction by nearly 2% after considering sentiment as a feature. So, we can say that sentiment is an important distinctive attribute of a sentence which should not be neglected.

6.4.1 Computing the sentiment

Vader was proposed in [23] as a sentiment analyzer tool for social media text. We use this to annotate English subtitles since phrases in social media could be treated similar to subtitles. For a specified input, Vader returns a dictionary of four scores namely positive, negative, neutral and compound. The compound score is computed by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalized to the range -1 (most extreme negative) to +1 (most extreme positive). This is the most useful metric if you want a single uni-dimensional measure of sentiment for a given sentence. This is referred to as the “normalized, weighted composite score”. We used ipublia[2] to annotate the German subtitles as Vader doesn’t support German text. Each subtitle was then classified as positive, negative or neutral.

6.4.2 Importance in translation systems

Considering the initial approach, the major issue we face here is that machine translators are poor propagators of sentiment in general. We aim to tackle this problem. This is one of the reasons why

---

the transfer of sentiment while translating is of great importance in multiple tasks and why we need evaluation metrics that take this into account. This is an important task since globalization has lead to a large use of translation systems which are, at times, the only means of communication between sets of people or even organizations. One of the most important aspects of communication is the emotion and the positivity/negativity of the context. One would certainly not want to say something nice and polite and it gets translated to something rude and impolite in the target language unknowingly. The worst part is that the first party doesn’t even realize that something like this has happened during the process of translation. Consider the following example sentences taken from the dataset

Reference 1 - It was quite stupid of you to run away again and to believe I wouldn’t find you.
Translation 1 - A little stupid of you running away again and believing that I would not find you.
Reference 2 - When I was first taken from my village I hated you with all my heart
Translation 2 - When they took me out of my village I thought I would hate you forever
Explanation - In the above-mentioned references, we can see that the word “quite” was misinterpreted as “little” and the word “hated” was misinterpreted as “thought I would hate” respectively leading to a drastic change in sentiment. The differences might seem subtle but it actually changes the meaning/opinion to a great extent.

6.4.3 Use cases

Situations where we need to use automated machine translators in business contexts etc must ensure that the sentiment is translated appropriately and there are no mistakes where the source statement could be intending something positive but it ends up getting translated to something negative or vice versa.

Another example could be where someone is learning a new language with the help of an automated machine translator and they learn greeting people in formal situations. In this case, if sentiment is not carried across properly, we might end up greeting them informally.

Considering product/movie reviews, we notice that many a time websites are translated for users and when these reviews get translated, it becomes an issue if the sentiment is not carried across correctly. Since reviews are often quite personal and are filled with emotion, being unable to carry this across well during translation would mean that we might end up incorrectly representing the reviews which is something we would like to avoid.

6.5 Proposing SEMMT

In order to work towards solving the issues presented in earlier sections, we have come up with a new metric called the SEMMT score that we use to evaluate machine translation models with in order to ensure they capture the sentiment well during translation. The exact definition of the metric and formulation details will be discussed in the Section 6.5.2.
Given a translation model $X$ that we wish to train ensuring that the sentiment is carried across well, we start training the model (using any of the existing standard training methods such as back-propagation, etc) and then use the SEMMT score at the final layer which could be used to verify how well the model performs.

In order to test different models and see which one performs better comparatively, we take the bilingual test data that we have at hand and then pass it through both the models to get corresponding outputs. We then evaluate the SEMMT scores for each of them and the one with a higher SEMMT score would be the model that performs better at maintaining both the sentiment and the quality of translation.

We wanted to mitigate the disadvantages of the popular existing metrics such as BLEU and at the same time take sentiment into consideration while evaluating a machine translator. BLEU has certain downsides such as it does not consider the meaning, does not directly consider sentence structure, does not handle morphologically rich languages well, etc. Some of the disadvantages of ROUGE include, it does not give importance to a sentence of candidate summary if the sentence does not have any matching word pair with its references summary sentence (it assigns a score of zero in this case) etc. Thus, we wanted to take an approach involving sentence embeddings.

### 6.5.1 Skip-thought cosine similarity

We use the notion of skip-thought vectors to generate sentence embeddings and use cosine similarity as mentioned in [30] to compare the reference and translations. The underlying assumption here is that any information present in a sentence which leads to a better reconstruction of the neighbouring sentences is also the essence of the sentence. These skip-thought vectors not only take word order into account but also take the ordering of sentences into account. This allows it to encode rich information into the embedding. We used the code provided in [52] to compute the skip-thought cosine similarity ($\text{SkipT} \_\text{CS}$). The representations of semantically similar sentences are closer. For example, the representation of “he ran his hand along her hair checking for any hidden microphone.” and “he checked her hair for a microphone, but could not find any” are very close by.

### 6.5.2 Formulation

Once the subtitle pairs are segregated into the three classes, we translate the German subtitles to English using Google\(^3\) and the IBM Watson\(^4\) machine translators. Now we have pairs consisting of the original English subtitles and their corresponding translated ones. We then assign a sentiment score to each of the subtitles using Vader. Here $s_0$ and $s_t$ are the sentiments of the original and translated subtitles in each pair respectively. Upon computing the value of skip-thought cosine similarity, the result ranges between 0 to 1. Continuing further, the sentiment value annotated by Vader varies from -1 to 1. This

\(^3\)https://translate.google.com/

\(^4\)https://www.ibm.com/watson/services/language-translator/
would mean that the absolute difference in the sentiment scores would range from 0 to 2. We then reshape this to 0 to 1 by dividing it by 2. This would ensure that both the ranges are consistent.

\[ SE\text{MMT} = \frac{\text{SkipT}_{\text{CS}}}{1 + \frac{\text{abs}(s_0 - s_t)}{2}} \]  

(6.1)

The main aim of SEMMT is to penalize translation scores where the sentiments of the original and translated sentences are far apart and provide a higher score when the sentiments are closer to each other as shown in 6.1.

6.6 SEMMT Evaluation

A metric is deemed right if it correlates well with human judgement i.e giving scores just as any other human would. In order to check if the proposed SEMMT score correlates well with humans, we manually annotated 1,000 sentences. We then calculated the correlation between human and SEMMT scores and compared it with the correlation score between human and the skip-thought cosine similarity scores. We can see a huge improvement of 41.2% after we take sentiment into consideration as shown in Table 6.1. Details regarding how the manual annotation has been done is explained below. We used Kendall’s tau coefficient mentioned in [27], which is a measure of concordance between two paired variables to calculate the correlation. This rank-based correlation may be preferred over the standard correlation when the underlying data does not have a meaningful numerical measure, but it can be ranked or when the relationship between the two variables is not linear. Hence, this is an apt statistic measure of correlation in this particular scenario.

6.6.1 Annotation Details

We had mono-lingual annotators annotate the subtitles. Each subtitle was annotated by five annotators. They were shown the original English sentence and the translated English sentence. They had to give two scores, a translation score (t) which rates how well the sentence has been translated and a
sentiment score (s) which rates how well the sentiment has been retained after translation. We asked them to provide scores ranging from 1 through 3. The final translation and sentiment scores are the average of the scores given by all five annotators. We calculate the manual SEMMT score \(\text{semmt}_m\) from the two scores given by annotators using equation 6.2.

\[
\text{semmt}_m = \frac{t-1}{3} + \frac{s-1}{3} 
\]

This ensures that the provided scores are normalized to the range 0 to 1 which is similar to the range of scores that our SEMMT formulation provides for machine translators.

6.6.2 Results

The function, represented in equation 6.1 provided an increase of 41.2% as shown in Table 6.1. We can observe a large increase in the case of IBM translations mainly because translations made by Google preserved the sentiment well compared to IBM translations. Below are examples from the dataset, seeing which we can understand why there is a huge increase in the correlation with IBM translations.

**Reference 1** - I think it’s only natural if you resent me a little.
**Google** - It would only be natural if you react a little negative.
**IBM** - It would be only natural if you respond.

**Explanation** - As we can see, the word “resent” means “disliking” which has a negative sentiment. The Google translator has preserved the sentiment by using the word “negative” whereas the IBM translator did not preserve the sentiment well.

**Reference 2** - They’re afraid because they subvert every great thing ever discovered.
**Google** - They are afraid because they question any great development.
**IBM** - They are scary because they ask every major development.

**Explanation** - In the above example, the word “afraid” means “fear”, which has a moderate negative sentiment. The Google translator has preserved the sentiment by using the same word “afraid” whereas the IBM translator used the word “scared” which means “extremely frightened” making it more negative than the reference.

**Reference 3** - Oh I coped with your world for 40 years and rather successfully at that.
**Google** - I’ve mastered her world for 40 years and I’ve been pretty successful.
**IBM** - I have 40 years and I was quite successful.

**Explanation** - In the above example, the word “coped” has been preserved in the Google translation by using “mastered” whereas the IBM translation skipped that part.

**Reference 4** - It’s only the last two or three years he started this drinking.
<table>
<thead>
<tr>
<th>Correlation</th>
<th>IBM</th>
<th>Google</th>
</tr>
</thead>
<tbody>
<tr>
<td>SkipT_CS vs manual</td>
<td>0.0827</td>
<td>0.2791</td>
</tr>
<tr>
<td>SEMMT vs manual</td>
<td>0.1168</td>
<td>0.2819</td>
</tr>
<tr>
<td>Improvement</td>
<td>+41.2%</td>
<td>+0.03%</td>
</tr>
</tbody>
</table>

**Table 6.1** Increase in values after considering sentiment

**Google** - He only started drinking two years ago.
**IBM** - With the drink he started just two three years.

**Explanation** - In the above example, we can see that the main gist of the sentence has been transferred very well in the Google translation but the IBM machine translator could not manage to do the same.

### 6.7 Summary

To conclude, we believe that our work would allow us to device models that are closer to humans in the way they capture the sentiment of source phrases during translation which would prove to be useful in order to come up with more human like machine translators. Along with this, we also mitigate other issues such as traditional methods of evaluation using BLEU scores often used human translations as one of the references. There are a number of issues with this kind of an approach as we move forward. One of them is that as the amount of data that needs to be translated by humans increases in quantity, the quality drops as multiple user studies have shown in the past. They often end up making mistakes that we cannot afford to have. Another issue is that conducting human translation on large amounts of data is just not practically possible as we start scaling up our systems. Hence we believe that once we come up with models that we know perform well on sentiment transfer (SEMMT scores), we can go ahead and use them as baselines which would provide us with the target translations that we can use as references to train further models. We could always augment this knowledge with human results in order to further refine the system in place.
Chapter 7

Conclusions and Future Directions

Sentiment is a very active area of research in the field of Natural Language Processing. People often consider sentiment (in terms of positive or negative) as the most significant value of the opinions users express via social media. NLP for speech analysis, combined with a powerful social media monitoring strategy, organizations can understand customer reactions and act accordingly to improve customer experience, quickly resolve customer issues and change their market position. With the widespread use of social media, the need to analyze the content that people share over social media is increasing day by day. Considering the volume of data coming through social media, it is quite difficult to do this with human power. Therefore, the need for applications that can quickly detect and respond to the positive or negative comments that people write is increasing day by day.

In this thesis, we start by predicting rating of a video by taking summary as an additional feature. We show that there is a significant improvement in the accuracy due to this. We came up with a novel model to predict the rating by taking features from both video and text. We use the Inception-V3 model to extract features from the video and CNN to extract features from the summary. We then append the features and pass them to a fully connected network to predict the rating. We can conclude that text has valuable information which if used wisely can lead to good results.

Next in Chapter 4 we consider features taken only from the summary and try to achieve results close to the results we got when we used both video and text. We propose novel architectures which predict genre and rating of movies from the summary. We run multiple experiments in order to find out which architectures give the best accuracy. We take word, char and sentence embeddings and train them on CNN’s, LSTM’s, GRU’s etc. We also propose an ensembling model which takes in word, char and sentence embeddings at once and predicts the genre and rating. We also show how the proposed models are generalizable and work well with other datasets.

Then in Chapter 5 we go a little deeper and consider the sentiment of the reviews while predicting the rating. We use the same models as mentioned in earlier. We show an increase of 2% in the accuracy after considering the sentiment. We can see that sentiment is a valuable feature which can help increase the accuracy of rating prediction. We managed to achieve good results even though Telugu is an agglutinative language which makes the task a challenging one.
Moving on to Chapter 6, we propose a metric called SEMMT which takes sentiment as a feature to score a machine translator. A machine translator gets higher score if sentiment is preserved after translating. We got an boost of 41.2% in the correlation with humans which is huge.

There is a lot of scope for building upon this work in future. One future direction would be considering emotions because in reality emotions provide a richer set of information that addresses consumer choices and, in many cases, even determines their decisions. Because of this, Natural Language Processing for sentiment analysis focused on emotions is extremely useful. Other directions in which this thesis can be extended is detecting sarcasm, negation, word ambiguity, multi-polarity etc.
Related Publications

1. Varshit Battu, Vishal Batchu, Murali Krishna Reddy, Radhika Mamidi, April, 2019. **SEMMT: Sentiment based evaluation metric for Machine Translation.** The SIGNLL Conference on Computational Natural Language Learning (CoNLL 2019), Hong Kong. (Under review)

2. Varshit Battu, Vishal Batchu, Rama Rohit Reddy, Murali Krishna Reddy, Radhika Mamidi, December, 2018. **Predicting the Genre and Rating of a Movie Based on its Synopsis.** The 32nd Pacific Asia Conference on Language, Information and Computation (PACLIC 32), Hong Kong.


Bibliography


