Burp NLP: A Rick and Morty text analysis

Text Analysis

an der Fakultät für Wirtschaft
im Studiengang Wirtschaftsinformatik

an der
DHBW Ravensburg

Verfasser:	Anton Geiger
Ausbildungsbetrieb:	Festo SE & Co KG
Anschrift:	Ruiter Straße 82
	73734 Esslingen Berkheim
Dozent:	Prof. Dr. Oliver Sampson
Abgabedatum:	31.3.2025

Inhaltsverzeichnis

Abkürzungsverzeichnis

LSTM	Long Short Term Memory

Glossar

Abbildungsverzeichnis

Tabellenverzeichnis

Formeln

1 Einleitung

2 Hauptteil

2.1 Data Understanding

For our project we used three different datasets. The first one is a dataset is downloaded from kaggle where we find the transcripts until season 5 formatted in the following shape 2.1. As the transcript also holds information about the scenery there is not just spoken text included in there. We also find short descriptions about the sorroundings and scenery description. An example fo that is also visible in the first row of table 2.1.

episode no.	speaker	dialouge
1	Rick	stumbles in drunkenly, and turns on the lights. Morty! You gotta come on. Jus'... you gotta come with me.
1	Morty	rubs his eyes. What, Rick? What's going on?
1	Rick	I got a surprise for you, Morty.

Tab. 2.1: Rick and Morty Transcript Dataset

The other datasets are retrieved by self written web scraping scripts from the Rick and Morty Wikipage and the IMDB website. With the webscraping script we were able to create following table containing all episode descriptions provided in the Rick and Morty Fandom 2.2.

id	title	text
0	Pilot	n the middle of the night, an obviously drunk Rick bursts
1	Lawnmower Dog	Jerry complains that the family dog, Snuffles, is stupid
2	Anatomy Park (Episode)	It's Christmas, and Jerry tries to enforce the idea

Tab. 2.2: Rick and Morty episode descriptions dataset

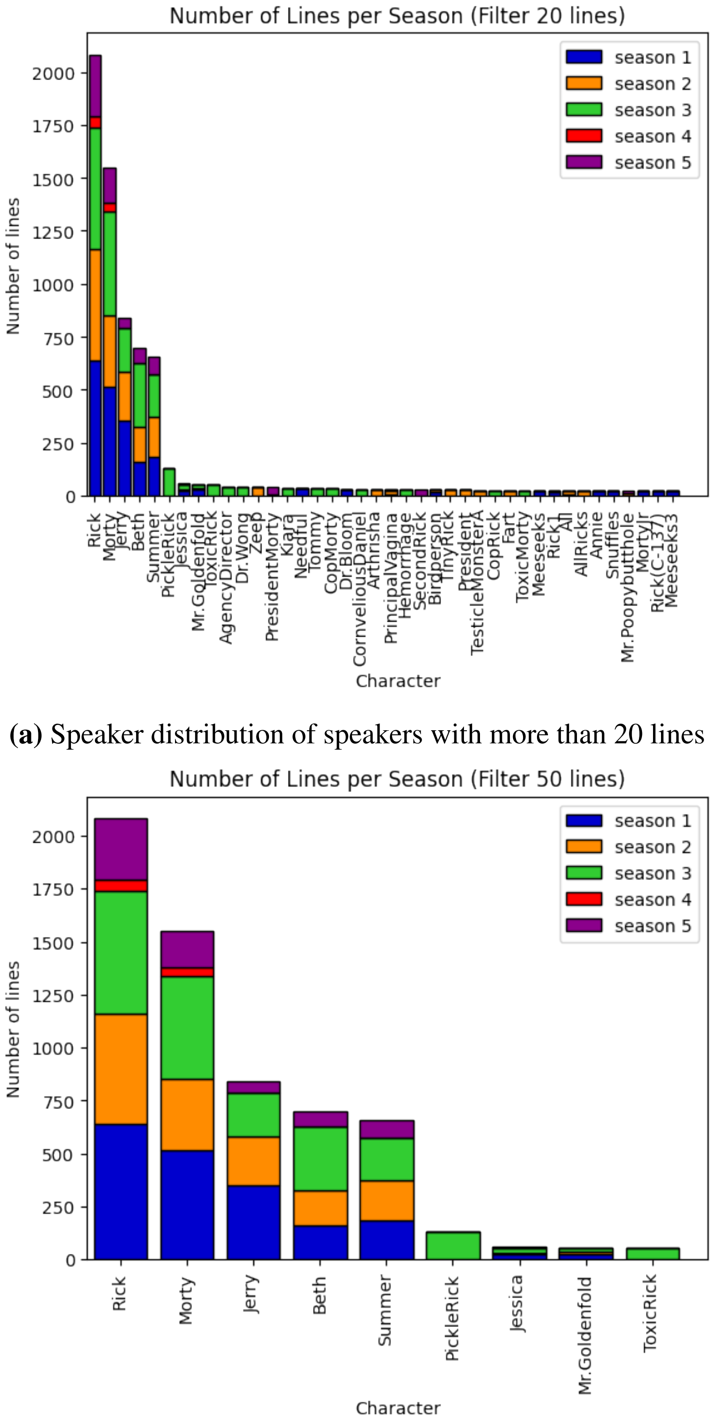
The last datasource stores information about the average IMDB rating per episode in the same format as provided in sample 2.3 where we also find the related season and title for each episode.

id	season	episode	title	rating
0	S1	E1	Pilot	7.9
1	S1	E2	Lawnmower Dog	8.6

Tab. 2.3: IMDB ratings per episode dataset

The rating score provided in the table is the average rating score on IMDB cacludated with every rating that was given to a episode. For the last two datasets we not just have the data until season 5 but we have every single episode until season 7 maintained.

Further analysis show that there are in general 970 different speakers. The charts 2.1 visualize the speaker distribution by counting the lines that each character speaks in each season. In general, there are a lot of different characters participating in the series but the family Smith including rick have by far the largest share when looking at the speaker distribution.



(a) Speaker distribution of speakers with more than 20 lines

(b) Speaker distribution of speakers with more than 50 lines

Abb. 2.1: Speaker distribution per lines

Surprisingly, the transcript dataset has some lacks. As we see in chart 2.3 that shows the number of dialogues per episode, the dataset is incomplete as there are a lot of empty episode starting from season 4. The background colors highlight the intervals of the different seasons along the whole series.

The chart 2.2 provides a clearer picture on how often which main charatcer speaks in detail.

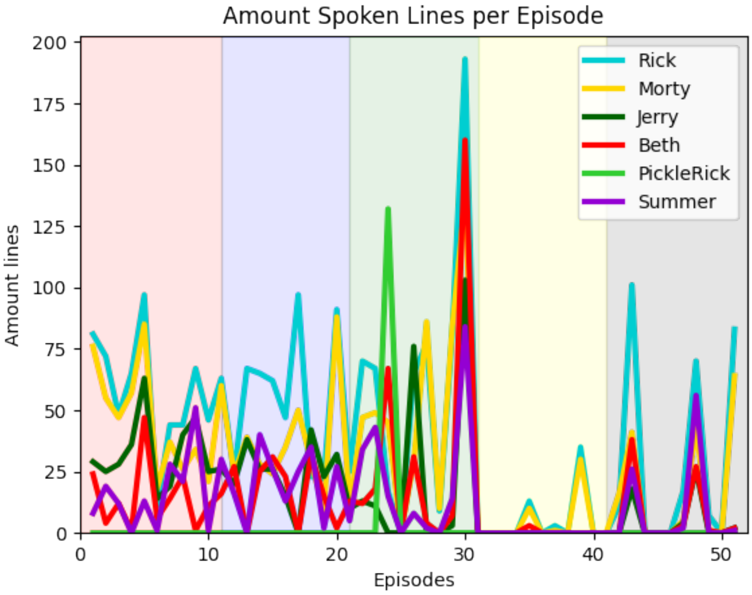


Abb. 2.2: Family Sanchez spoken lines in detail

In most of the episodes, Rick is the main character and holds the most shares in the speaking distribution. The other family members (Morty, Jerry, Beth and Summer) also speak in every episode but not that often as Rick. Side characters, like Pickle Rick often just have a large share in a few episodes as they often disapear after one episode.

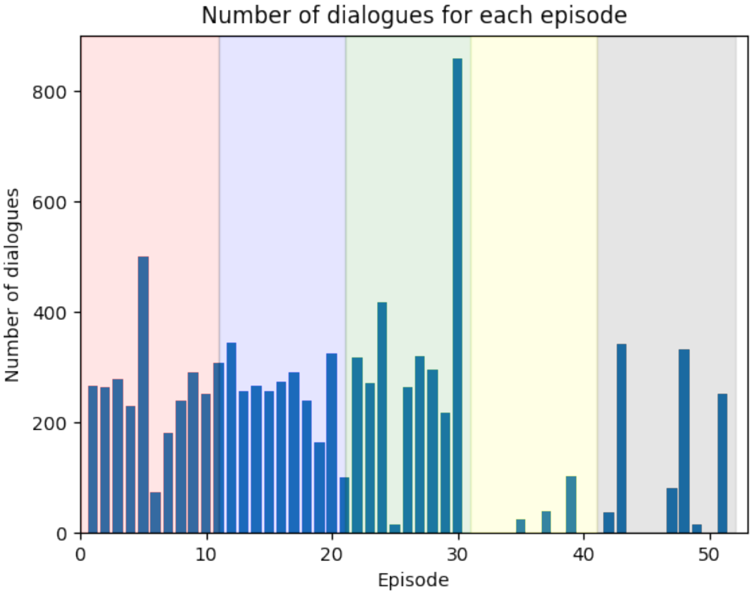
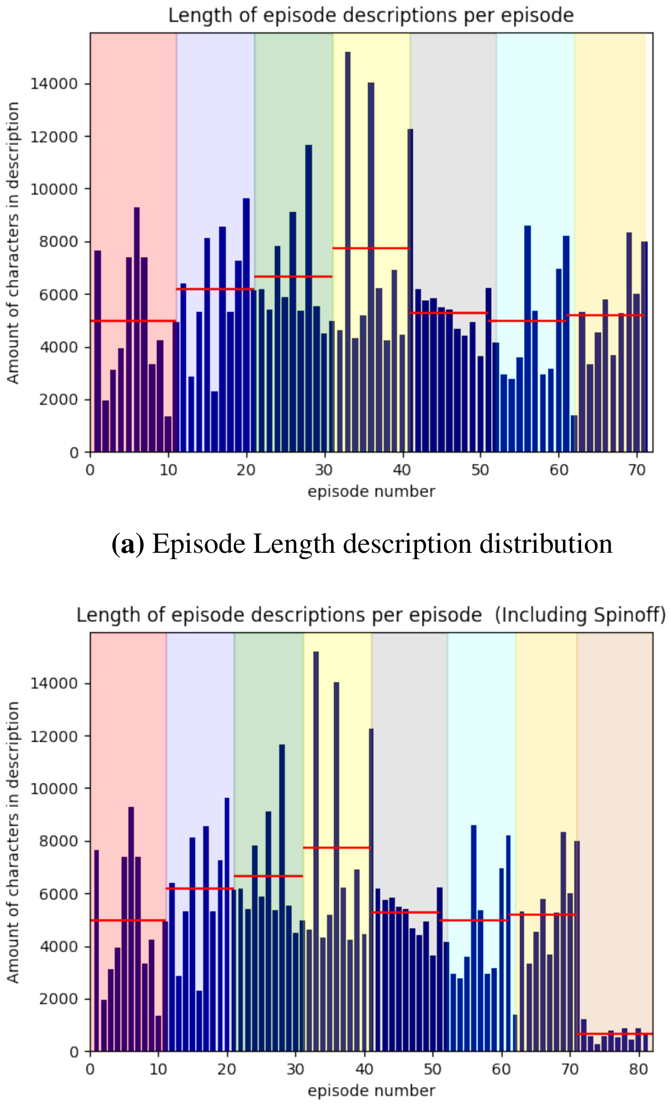


Abb. 2.3: Episode distribution

Looking deeply at the description dataset, we see two charts in 2.4 which compare the length the episode description for each episode. Surprisingly, the descritptions from episode 72 until 81 have a way shorter description than the episodes before. These episodes do not belong to the normal Rick and Morty series as they are spin-off episodes. Therefore, we now exclude those episode in further analysis. The red lines show the average description length per each season. We see that the description length really varies a lot for each season. As the description of episode 33 holds more than 3 times the amount of chars compared to episode 34. Another fact that is visible is that starting from season 5 there average number of chars in the description seems to stay on a similar level. That is different compared to the first 4 seasons where in each season the average length of the description increased.



(a) Episode Length description distribution

(b) Episode Length description distribution (Including Spinoffs)

Abb. 2.4: Length of episode descriptions per episode

The last analysis step was an analysis regarding the top 100 most spoken words in the transcript. We used a wordcloud to visualize those in an appropriate way. The results can be seen in 2.5. After removing the character names, we find there colloquial language terms like ’yeah’, ’gonna’ in the wordcloud. That is why we can assume that the spoken english is not that

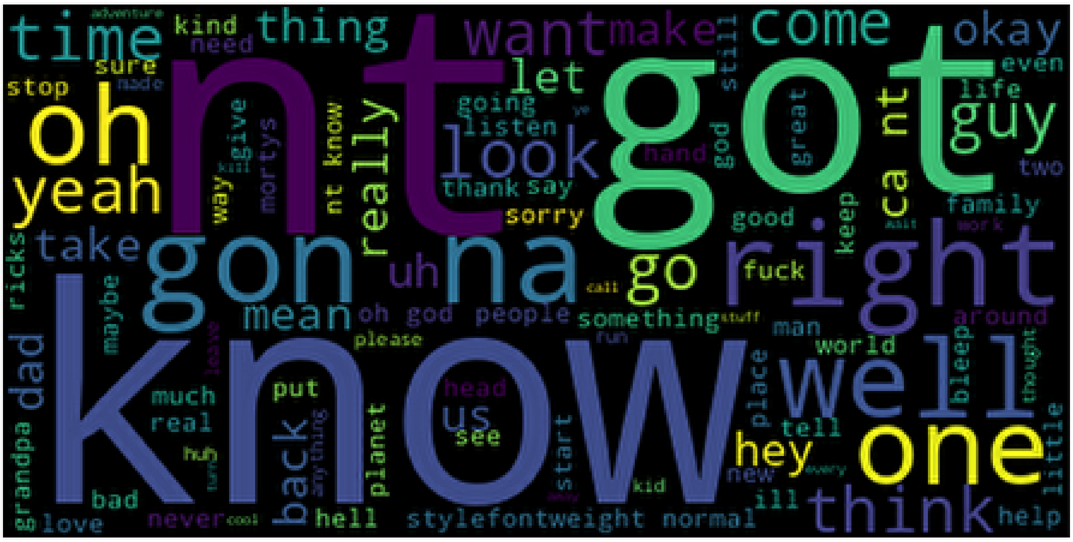


Abb. 2.5: Word Cloud transcript

advanced.

2.2 Data Preparation

Here, we describe the procedure we followed to clean and prepare our data. To achieve this, we used python libraries like nltk or spacy. To clean and unify the data, we removed all nonspace characters, and set every character to a lower character. As the rick and morty characters talk to each other with colloquial langaue, we removed some contractions like 'ain't' with is not. Furthermore, there are some html tags in the dataset that were removed in the data preparation phase.

Using the spacy and nltk library, the text preparation pipeline consists of the a custom stopwordremover and a custom stemmer which stems every single token in a sentence but ignores the character names as we do not want to change them. Doing this, we used the Porterstemmer. Before that, the sentences were tokenized using the small en web model spacy provides. To see the results, the first dialogue datapoint

stumbles in drunkenly, and turns on the lights. Morty! You gotta come on. Jus'... you gotta come with me.

turns into

stumbl drunkenli turn light morty got tocom jus got come

2.3 Topic Modeling

In this section we want to examine the Rick and Morty Series regarding the topics adressed in the first five seasons. To achieve this, we are looking at the transcripts and all of the episode descriptions, where we excluded the names of the main characters as they are not really important

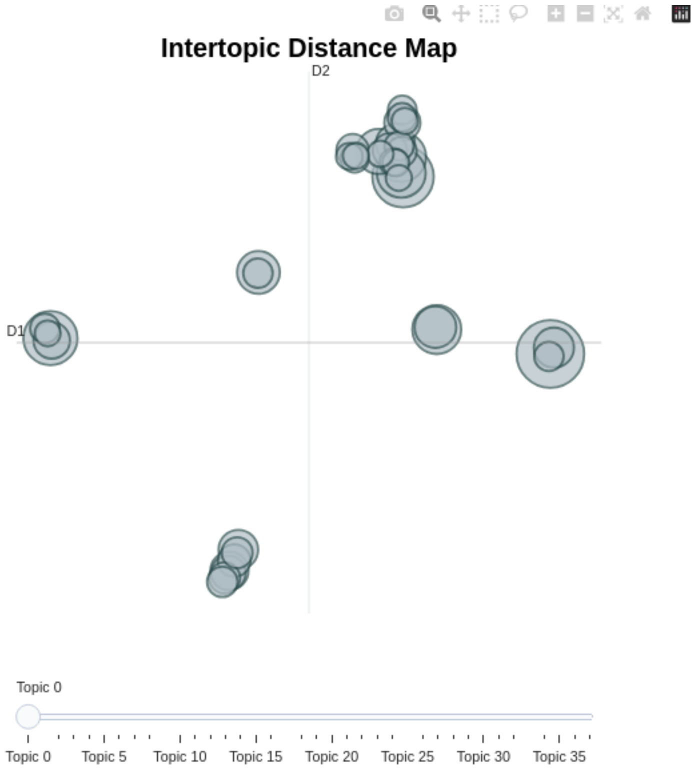


Abb. 2.6: BERT Topics for Transcrip (all seasons)

thematic topic.

Topic Modeling Transcript

At first, the topics where analysed based on the transcript in the first 5 seasons. As the dataset contains roughly 9.000 datapoints,we use BERTopic Model to find out the most relevant topics. For using the BERT Topic model, we downloaded the all-MiniLM-L6v2 embedding model to embedd the text. The embedded text then was clustered by the hierarchical HDBscan cluster algorithm using the eom cluster selection method. The Topic Map results are showcased in the image 2.6.

Whith the minimum topic size is set to 40 words, the model created 37 different topics. As in the figure provided, we see that most of the bubbles overlap. Therefore there would be the possibility to decrease the number of topcis consindering that this would leed to more imature topics. Analysing the content of the topics, we get on the left side topics that can be summarized with the keryword family. There we find terms like dad, mom, grandpa and so on. The topics located on the bottom of the image mostly refer to general information about the adventures that Rick and Morty perform as there are terms like adventure, portal gun and treasure. Especially the word portal gun is a really important word as it is a key component of the series. Most of the other topics can be summarized by the speaktng habits each character has. As there are topics where there are slang words like ’wubabdubadabab’, ’geez’ , ’oh’ or ’crap’. Surpisingly there is one topic located next to those slang words containing information about the planet system. This topics holds words like pluto, planet.

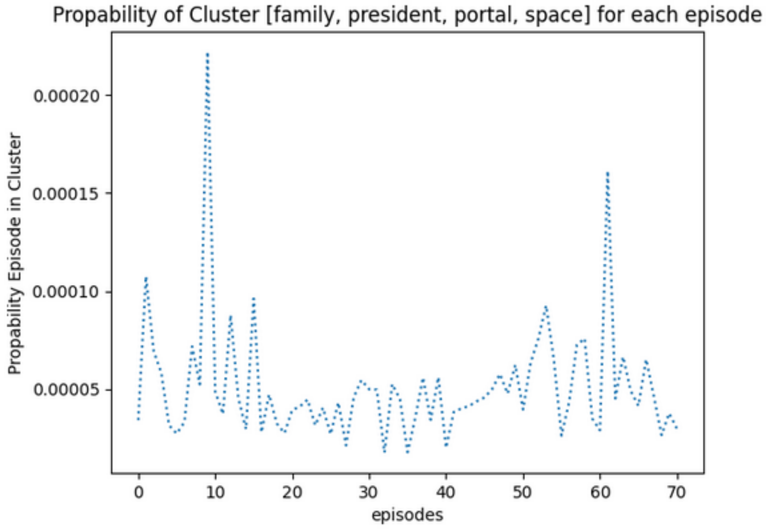


Abb. 2.7: Topic relevance Family Portal Gun over time

In 5.1, we see a visualization of the most important topics and its words. It turns out, that the most important topic are slang words like ’oh’, ’man’. Topic 2 and Topic 3 show that the series is about a human family. Topic 6, 11 and 19 contain more information about the content of the series as there are a lot of terms referring to space and murdering activities that Rick and Morty are exploring along their adventures in space.

Topic Modeling based on Episode Descriptions

As the dialouges of the series are full of abbreviations and colloquial language without a lot of content, we wanted to have a closer look at the topics by looking at all episode descriptions. This time we analysed the topics with Latent Dirichlet Allocation (LDA) as there are way less datapoints than in the transcription dataset. To setup our LDA model we create an Document Term Matrix (DTM) containing all words and documents. Another dictionary maps the words with a given id. Furthermore, we limit the amount of topics to 50 so that there will be less topic in the end than the number of episodes. In the series, the episodes do not really depend on each other. The following topic analysis can lead to the same conclusion as a lot of the created topics just contain information about the content of one or a few episoded. In the chart 2.7 we see the relevance of the topic containing family and portal gun along all episodes.

This topic plays a significant role in all episodes as ’family’ and ’portal gun’ are terms that are frequently used in a lot of episodes throughout the series.

As already mentioned, most of the other topicjust refer to a few or one single episode. This is the case for the topic titanic party where jerry and beth attend a party that recreates the titanic drama. As the chart 2.8 shows, this topic holds just information that is is mainly present in one episode.

The other spikes that are showcased in the chart can be explained as the side character ’Abradolf’ ’Lincoler’ plays a quite important role in other episodes as well.

Another example for a similar distribution is the topic about the spaghetti planet and its citadel,

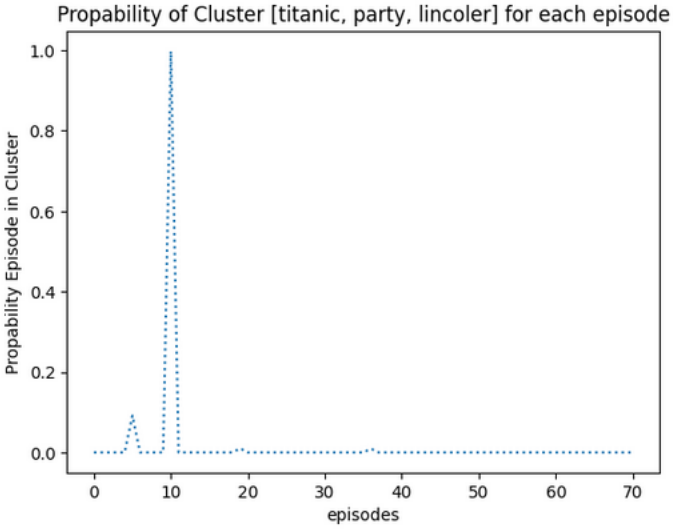


Abb. 2.8: Topic relevance Titanic Topic over time

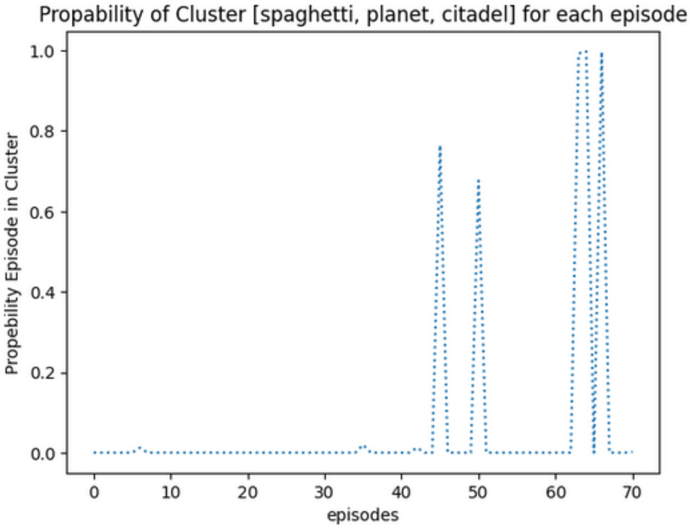


Abb. 2.9: Spaghetti an Citadel over time

which is analysed in chart 2.9.
The fact that most of the topics differ a lot from each other, can also be seen here 5.2. It is remarkable that a vast amount of the topics are distributed across the chart while just a handful topics seem to overlap. Furthermore, the size of the bubbles indicate the amount of documents where the topic is part of. As a lot of the bubbles seem quite small this could mean that there the topics mostly relate to a few episode descriptions.

3 IMDB Rating prediction

In this section our goal was to perform a modeling to predict the IMDB ratings based on the provided episode descriptions. Therefore, we compare different strategies and architectures to solve this complex problem.

Our hypothesis is that there are some important words or characters like ’birdperson’ that might have an impact on the IMDB rating as some characters or planets in the Rick and Morty universe are more popular than other.
Creating a perfectly fitting model is challenging, as the episode descriptions are written in neutral style and the features that determine whether an episode is liked or not can differ from person to person. Furthermore, there are also visual effects and music elements such as the ’get Schwifty’ or snake jazz song that might lead to a higher IMDB rating. Therefore, we expect that our model will tend to not be the most accurate one.

Vector Embedding Approach
As the neural network cannot work with strings as we human, we have to represent the given text as numeric representations. To achieve this, our intial approach used the Word2vec library to generate a vector embedding for all the words that are that appear at least 5 times in all descriptions. By that, we were able to represent most words in the Rick and Morty corpus as an vector of 100 values.
These 100 values also defined the input shape of our neural network which was designed for classifying the episode descriptions. To represent whole tokenized sentences into a vector containing 100 dimensions, we tokenized all of the training data and calculated the average vector of each episode description. As a result, each episode description can be represented as one single vector of 100 dimensions. With that, we trained the neural network. If a token was not part of the Word2vec vector embedding list, the token will be skipped. For example, the string ’DHBW Ravensburg’ would lead to a NaN and the string ’Rick DHBW Ravensburg’ would produce the same result as the string ’Rick’, since the unkown words are like DHBW in this case, is ignored.

The test results are displayed in the following chart 3.1. The green bars represent the episodes having a positive ratings while the blue ones have a lower rating. As we did not set a thresshold yet, we can see the test result visualized in a bar chart.
As expected, the neural network struggles to distinguish between those two classes. In general, it seems that the models even predicts the complete opposite, as the blue colored bars were predicted higher as the the green ones. The calculated test accuracy based on the results without an thresshold was 33%.
Setting the thresshold to 0.5 we find following confusion matrix as an result 3.2. The confusion matrix also leads to a similar results as there are more wrong predicted values than right ones. Letting us know that this model definetly does not solve the problem.
LSTM Approach
As the first neural network had some difficulties to examine the complex relations between the words, we also implemented the Long Short Term Memory (LSTM) architecture to achieve more reliable results. Here, we used the same Word2vec embedding as before. We trained our

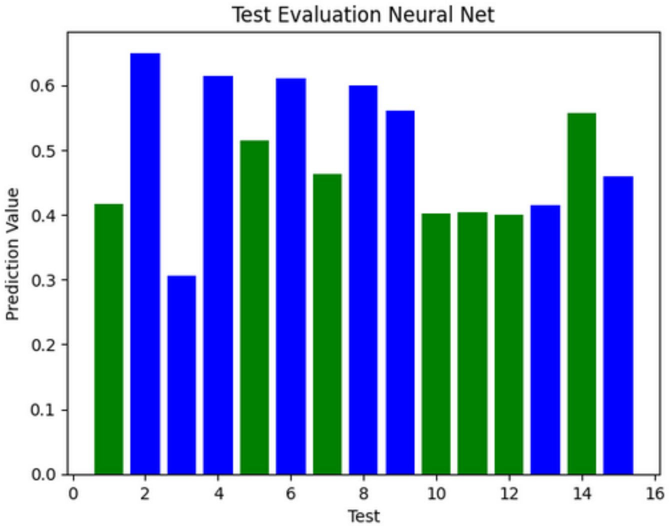


Abb. 3.1: Neural Net Test Evaluation

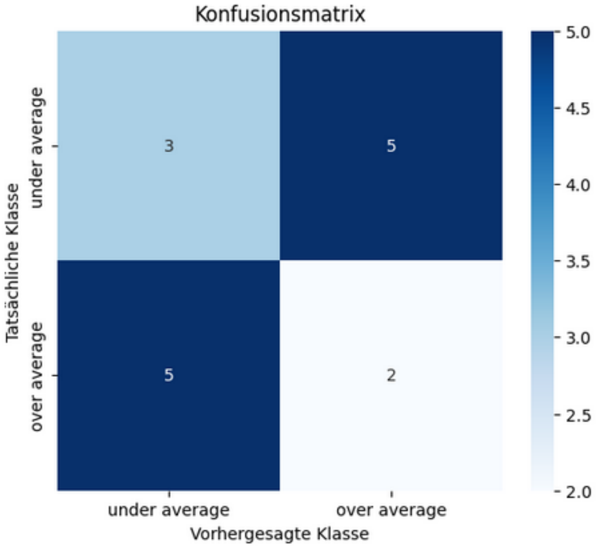


Abb. 3.2: Neural Net confusion matrix (thresshold = 0.5)

own Word2vec embedding model due to the fact that there are a lot of Rick and Morty unique terms like ’birdperson’, ’portal gun’ in the descriptions. Unlike the previous neural network, which computed the average vector for each sentence, we instead used the first 500 vectors in a sentence to numerically represent the entire text. The results are displayed in the chart3.3.

As the LSTM Classification leads to more variety in text prediction, it still had some problems figuring out which texts are above and under 8.2 rating. As the confusion matrix displayed in 3.4 shows, the model is better in distinguishing the classes than the first approach, as there are more correct prediction than wrong ones.

LSTM Regression

In this section, we aim to predict the IMDB rating not by classifying the episodes as 1 or 0 but by directly predicting the IMDB score. To achieve this, we used the same architecture as in the classification task but changed the lossfunction and we additionally normalised the word2vec

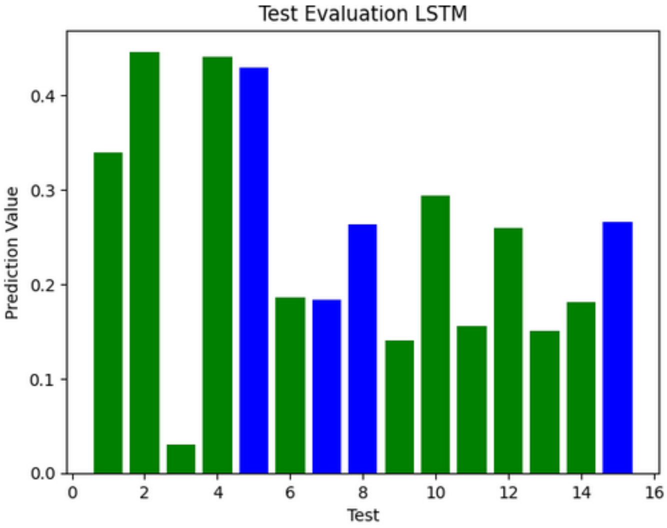


Abb. 3.3: LTSM Classification

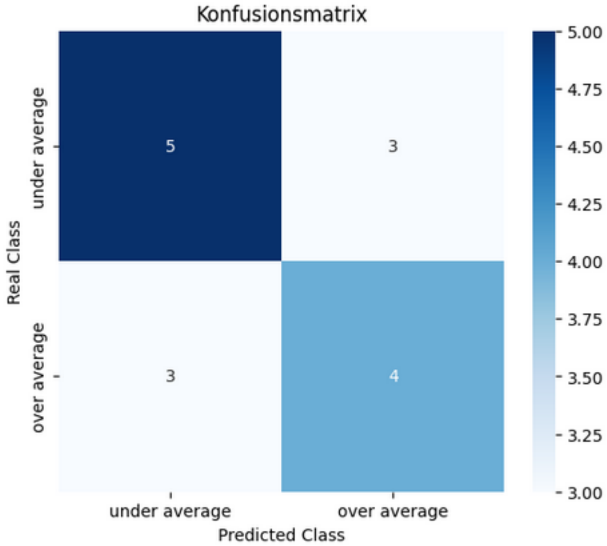


Abb. 3.4: LTSM Classification confusion matrix (treshhold = 0.4)

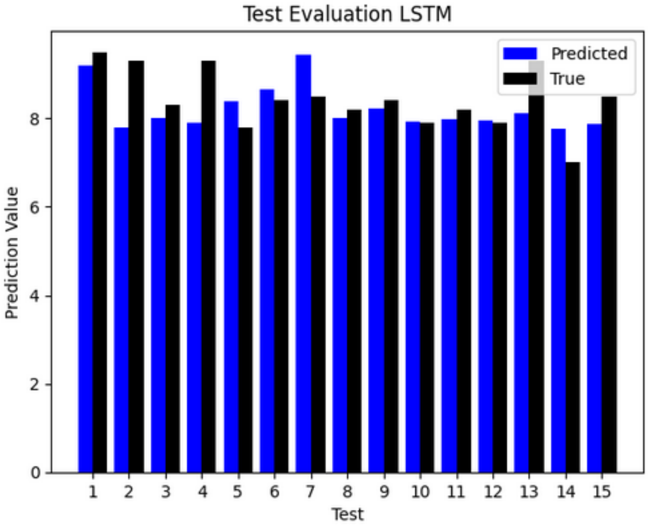


Abb. 3.5: Test Results Regression

embedding vectors in the matrix.
As the validation error starts to increase, we set 25 as our default epoche size.
Another Cross Validation got an average error of 0.47 5.1, meaning that, on average, our model's predictions deviate by approximately 0.5 rating points from the true IMDB scores. An analysis of the test results is displayed in the chart 3.5. There we, see the predicted IMDB ratings in blue and the True ones in black. In general, we see that the model predicts valid results, as none of the predicted IMDB ratings were above 10. In general, the model seems not to bad, as most of the deviation of prediction and true values are not very large.

3.1 Speaker detection

Our last model should predict a given speaker on a given text dialogue. Therefore, we just use the family sancez because speakers because of the missing character lines. This would make it hard to determine speaking traits. As already seen in chart 2.1, the dataset is imbalaced which can cause some problems. In order to get rid of this imbalaced dataset, we decided to duplicate the amount of spoken lines of jerry, beth and summer.
For this classification problem, we used the small DisitllBERT transformer as a base model consisting of the DistillBERT Tokenizer and a Sequence classifier. As the test error still decreases whith 3 epochs, we fine tuned the model with 3 epochs. Doing this, we created a model that had a 65% accuracy on the test set. There we have to keep in mind that there might be duplicates in the test set due to the duplication in the first step.
For example the Input "Burp"lead to following outout (Rick: 3.9 , Summer, -0.5, Morty -1.4, Jerry: -1.6, Beth: -1,9). There, the model was quite sure that propably rick would have said this in a conversation. This is also realistic as he often needs to burp after drinking beer. The overall results can be seen in following confusion matrix 3.6 and interpreted with the metrics in 5.2.

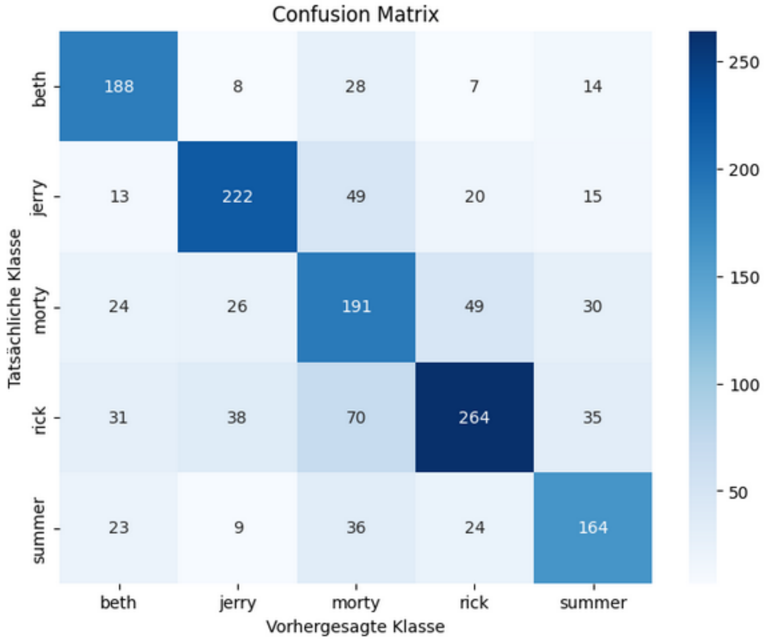


Abb. 3.6: Confusion Matrix speaker detection

4 Diskussion

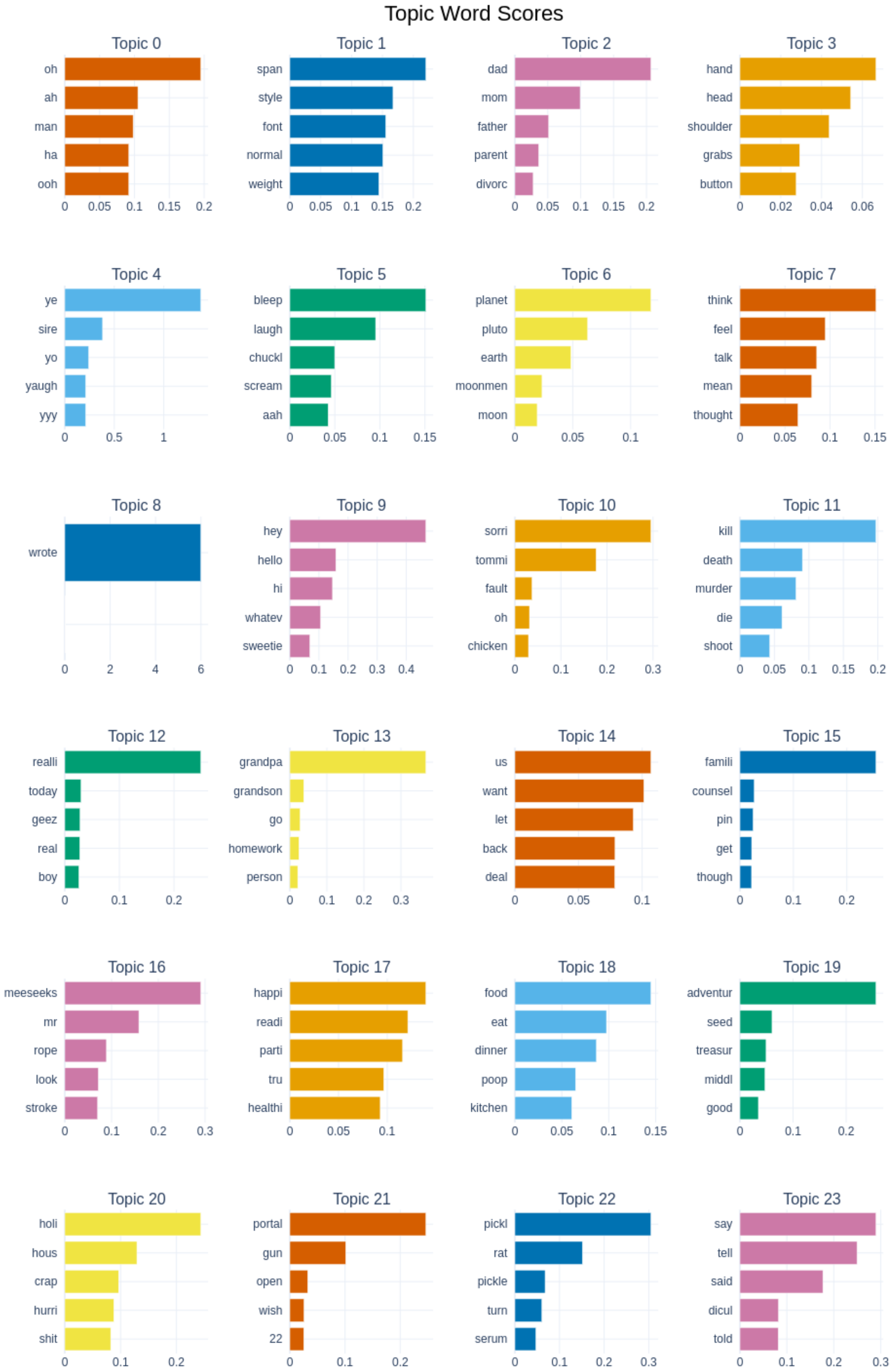
5 Anhang

mean average error
0.4996431767940521,
0.4830387234687805,
0.5242437124252319,
0.4312141239643097,
0.49884870648384094

Tab. 5.1: 5 Fold Cross Validation

	precision	recall	f1-score	support
beth	0.67	0.77	0.72	245
jerry	0.73	0.70	0.71	319
morty	0.51	0.60	0.55	320
rick	0.73	0.60	0.66	438
summer	0.64	0.64	0.64	256
accuracy	0.65	1578		
macro avg	0.66	0.66	0.66	1578
weighted avg	0.66	0.65	0.65	1578

Tab. 5.2: Metrics for speaker detection


Topic Word Scores
Topic 0
oh
ah
man
ha
ooh
0
0.05
0.1
0.15
0.2
Topic 1
span
style
font
normal
weight
0
0.05
0.1
0.15
0.2
Topic 2
dad
mom
father
parent
divorc
0
0.05
0.1
0.15
0.2
Topic 3
hand
head
shoulder
grabs
button
0
0.02
0.04
0.06
Topic 4
ye
sire
yo
yaugh
yyy
0
0.5
1
Topic 5
bleep
laugh
chuckl
scream
aah
0
0.05
0.1
0.15
Topic 6
planet
pluto
earth
moonmen
moon
0
0.05
0.1
Topic 7
think
feel
talk
mean
thought
0
0.05
0.1
0.15
Topic 8
wrote
0
2
4
6
Topic 9
hey
hello
hi
whatev
sweetie
0
0.1
0.2
0.3
0.4
Topic 10
sorri
tommi
fault
oh
chicken
0
0.1
0.2
0.3
Topic 11
kill
death
murder
die
shoot
0
0.05
0.1
0.15
0.2
Topic 12
realli
today
geez
real
boy
0
0.1
0.2
Topic 13
grandpa
grandson
go
homework
person
0
0.1
0.2
0.3
Topic 14
us
want
let
back
deal
0
0.05
0.1
Topic 15
famili
counsel
pin
get
though
0
0.1
0.2
Topic 16
meeseeks
mr
rope
look
stroke
0
0.1
0.2
0.3
Topic 17
happi
readi
parti
tru
healthi
0
0.05
0.1
Topic 18
food
eat
dinner
poop
kitchen
0
0.05
0.1
0.15
Topic 19
adventur
seed
treasur
middl
good
0
0.1
0.2
Topic 20
holi
hous
crap
hurri
shit
0
0.1
0.2
Topic 21
portal
gun
open
wish
22
0
0.1
0.2
Topic 22
pickl
rat
pickle
turn
serum
0
0.1
0.2
0.3
Topic 23
say
tell
said
dicul
told
0
0.1
0.2
0.3

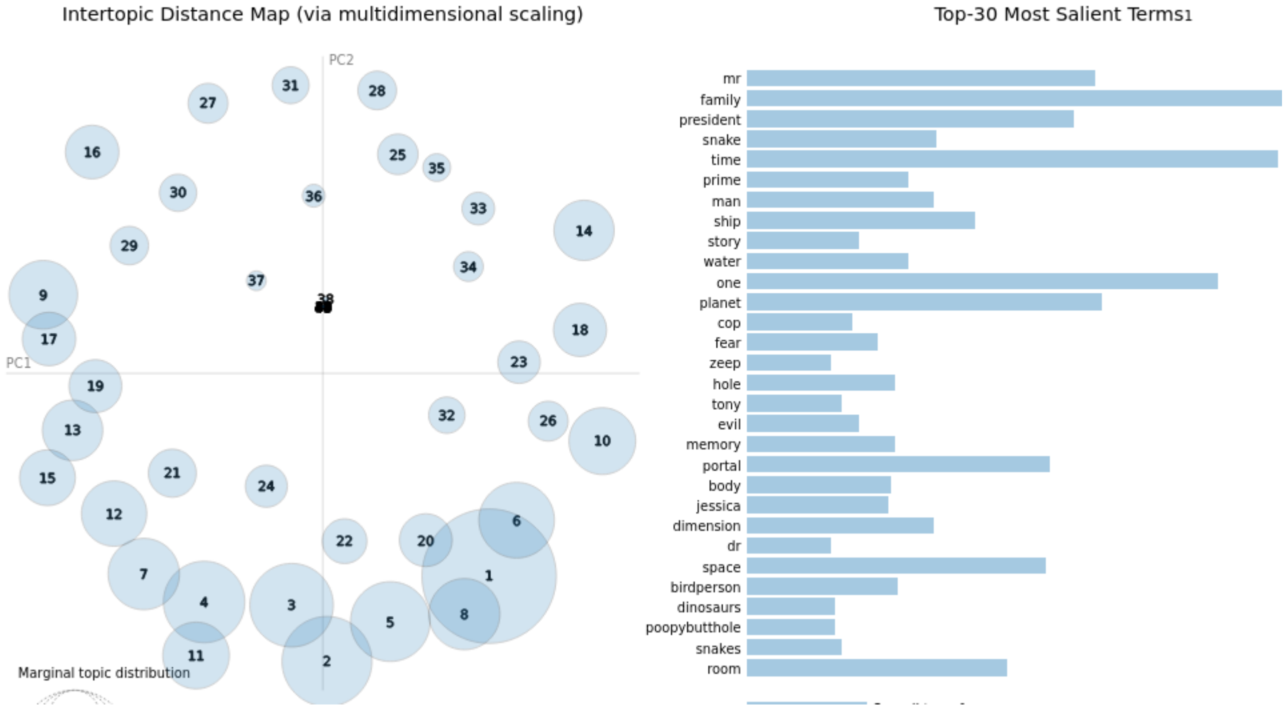


Abb. 5.2: All Topics LDA for description (all seasons)

Literatur

Selbständigkeitserklärung Anton Geiger

Ich versichere hiermit, dass ich, Anton Geiger, meine Seminararbeit

Burp NLP: A Rick and Morty text analysis

selbständig verfasst, die digitale Version mit der ausgedruckten
Version übereinstimmt und keine anderen als die angegebenen
Quellen und Hilfsmittel benutzt habe.

Ort, Datum

Unterschrift