Natural Language Processing
Phase №2: Data

Pooya Kabiri

Department of Computer Science

Iran University of Science and Technology

May 2021

Contents

List of Figures

List of Tables

1 Raw Data

1.1 Data Source

I used the Friends TV series transcripts, I found them on Kaggle website, In forms of `.txt` files for each episode, Including `228` total episodes for `10` seasons.

I used the Kaggle API package, which works from the command line for downloading files.

Link to Transcripts

1.2 Labeling

In a transcript file, each line is either a episode title, story author, dialogue line, or scene and gesture explanation.

The following example lines are from `Episode 17` of `season 7:`

```
   The One With The Cheap Wedding Dress
   Teleplay by:  Andrew Reich & Ted Cohen
   Story by:  Brian Buckner & Sebastian Jones
   [Scene:  Central Perk, Monica, Chandler, Phoebe, and Joey are there.
Monica is holding a piece of paper.]
   Joey:  Food?  Uh-huh gimme!  (She hands him the paper.)
   Phoebe:  Since when are you into swing music?
```

As you can see, each line of dialogue starts with the name of the charcter followed by a colon, so I processed transcript files line by line, and from this starting word, I labeled each dialogue to each of the 6 character classes, which are `MONICA, JOEY. CHANDLER, PHOEBE, ROSS, RACHEL.`

In each processing step, corpus for each class is saved in a separate `.txt` file with the name of the file indicating the class name. When reading the data, this file name is used as the class name.

2 Preprocess

2.1 Tools

- **Cleaning**: `python re` module, `python string` class(default), with `string.replace()` method.
- **Sentence & Word Splitting**: `python string` class(default), with `string.split()` method.
- **Lowercase Transform:** `python string` class(default), with `string.lower()` method.

- **Word & Sentence Tokenization:** `python nltk` module, with `sent_tokenize()` and `word_tokenize()` methods.
- **Contraction Expansion:** `python contractions` module.
- **Stopword Removal:** `python nltk` module, using English stopwords.
- **Lemmatization:** `python nltk` module, using `WordNetLemmatizer` class.
- **Stemming:** `python nltk` module, using `SnowballStemmer` class.

2.2 Unit

the unit of data is **Dialogue**, each dialogue can contain multiple sentences.

2.3 Downloading

The Kaggle API is executed using a bash script, named `download.sh`, which uses a Kaggle API key to download the dataset, and unzips the zip file.

2.4 Cleaning

For cleaning the data, I do the following procedure:

1. Stripping the `\n` and `\t` characters.
2. Removing the text inside parentheses, which is related to gesture and scene understanding, and is not a spoken dialogue, using Regular Expression matching, with `python re` package.
3. Replacing Hyphens(-), Double Quotation,("") Three dots(...), Double Dots(..) with white space.
4. Removing any non-ASCII characters from the corpus, using Regular Expression matching, with `python re` package.

2.5 Tokenization

I tokenize corpus for each class, based on word and sentences, and save the results in the corresponding files.

2.6 Steps

The preprocessing of data is done in 9 steps, as follwoing:

1. **Downloading**
2. **Cleaning**
3. **Lowercase Transform**

4. **Contraction Expansion**
5. **Word Tokenization**
6. **Sentence Tokenization**
7. **Stopword Removal**
8. **Lemmatization**
9. **Stemming**

Steps are done sequentially and in the written order.

2.7 Implementation

2.7.1 Code Structure: Modular

Each step saves the results in it's corresponding folder, with a separate file for each class, and the next step reads the data from the files generated by the step(s).

The main driver code checks if a step is already done, then in the procedure, that step is skipped and not done again.

The main `driver.py` scripts import codes from two sources:

- `preprocess.py`, which import functions from different files, each corresponding to a single preprocessing step.

  This file itself uses the functions in 9 different files, each one corresponding to a different preprocessing step:

  - `raw.py`: python file, in charge of downloading and unzipping the data, by running the `download.sh` bash script with `python subprocess`, for invoking Kaggle API, downloading the data, unzipping the data in `/data/raw/` directory.

  - `clean.py`: python file for reading each `.txt` file generated in the previous step, labeling the data, cleaning corpora, and saving it in two different shapes: separated by dialogue units (`/data/clean/dialogue/` directory), and joined corpus (in `/data/clean/corpora/` directory). In each directory, 6 files are generated each for one labeling class.
  - `tolower.py`: python file for Lowercase Transform of the previous step corpora, results are saved in `/data/tolower/` directory.
  - `contraction.py`: python file for Contraction Expansion of the previous step corpora, results are saved in `/data/contractions/` directory.

  - `word_tokenize.py`: python file for Word Tokenization of the previous step corpora, results are saved in `/data/word_tokenize/` directory.
  - `sentence_tokenize.py`: python file for Sentence Tokenization of the Contraction Expansion step corpora, results are saved in `/data/sentence_tokenize/` directory.
  - `stopword.py`: python file for removing the English Stopwords from the Word Tokenization step corpora, results are saved in `/data/stopwords/` directory.
  - `lemmatize.py`: python file for Lemmatization of the Stopword Removal step corpora, using `WordNetLemmatizer` from `nltk`, results are saved in `/data/lemmatize/` directory.
  - `stemmer.py`: python file for Stemming of the Stopword Removal step corpora, using `SnowballStemmer` from `nltk`, results are saved in `/data/stemming/` directory.

- `analysis.py`, which holds the required functions for the analysis parts of the data, as well as plotting.

2.7.2 Execution

The whole project can be executed with the driver file, by the following command :

```
python driver.py [preprocess options] [analysis option]
```

The options can be used in the following ways:

- `[preprocess options]`:

  This option is an integer, indicating the preprocess step required. You can find the step number as following:

  - **0**: Downloading and Unzipping raw data.
  - **1**: Cleaning Data.
  - **2**: Lowercase Transform.
  - **3**: Contraction Expansion.
  - **4**: Word Tokenization.
  - **5**: Sentence Tokenization.
  - **6**: Stopword Removal.
  - **7**: Lemmatization.
  - **8**: Stemming.

The options are cumulative, meaning that 5 means all steps until Sentence Tokenization. Each step is executed if it's result is not present, otherwise skipped.

If no option is provided, the default value for this option is 8, meaning executing all steps.

- `[analysis option]`

  This option is a string, indicating if the script should run the analysis part or not.

  By providing `--analysis`, the script checks the current preprocess status(stage), if it is sufficient for running the analysis (the required step for analysis is step 7, Lemmatization), runs the analysis script from `analysis.py`, otherwise runs the preprocessing method with `force` flag set to `True`, which removes all the preprocessed data, updates the raw data and runs each preprocess step fresh from start, until the required lemmatization step, which is required for the analysis.

  The result of the analysis is saved as a `.txt` file, names `analysis_report.txt` in the `/analysis/` directory, and as charts generated, saved in `/analysis/charts/` directory as `.png` files, each corresponding to one required analysis step.

Some examples of Execution Command:

Preprocessing until Word Tokenization, and running analysis:

```
python driver.py 4 --analysis
```

Preprocessing default (all stages - until Stemming), and running analysis:

```
python driver.py --analysis
```

Preprocessing until Stopword Removal, with no analysis:

```
python driver.py 6
```

Preprocessing default (all stages - until Stemming), with no analysis:

```
python driver.py
```

3 Analysis

3.1 # Units

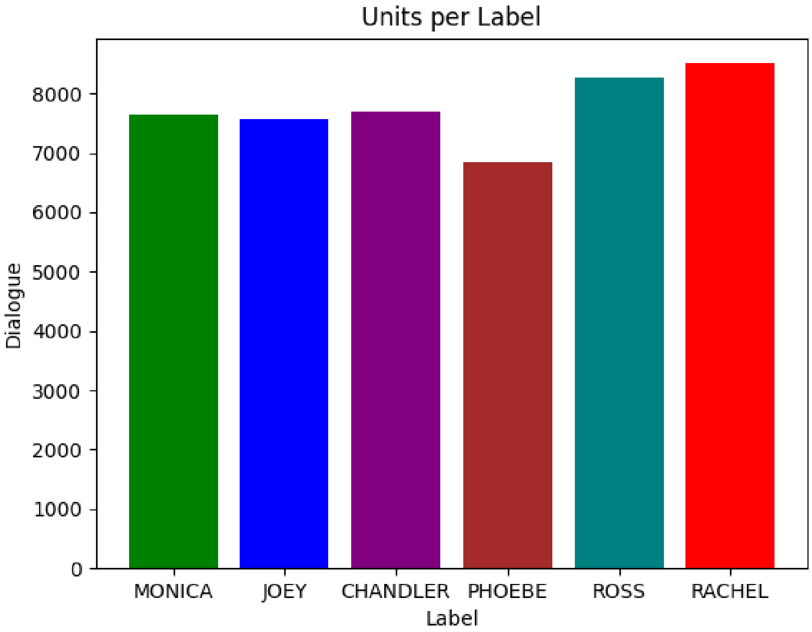

Figure 1: # Units per Label(Character)

	Monica	Joey	Chandler	Phoebe	Ross	Rachel
#Units	7651	7571	7686	6832	8262	8506

Table 1: # Units per Label(Character)

3.2 # Sentences

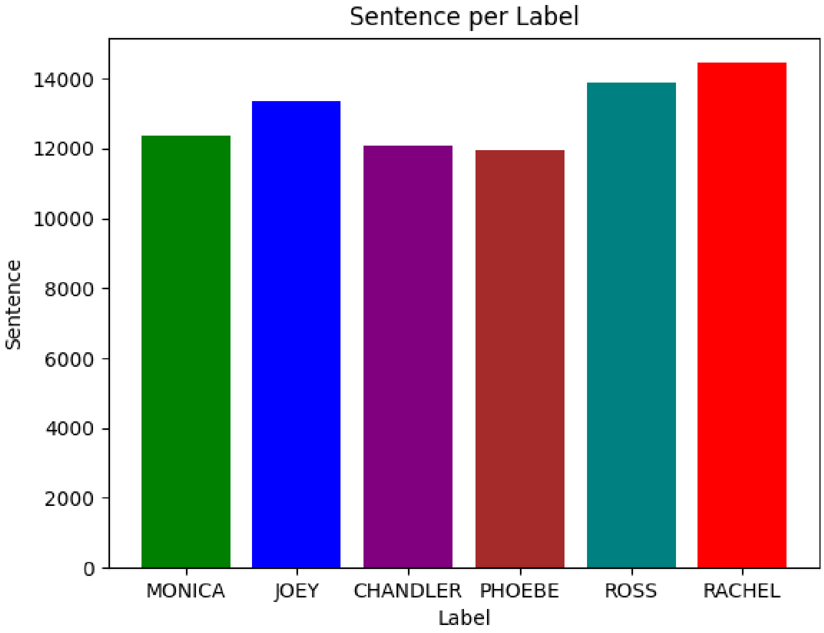

Figure 2: # Sentences per Label(Character)

	Monica	Joey	Chandler	Phoebe	Ross	Rachel
#Sentence	12370	13347	12064	11967	13873	14452

Table 2: # Sentences per Label(Character)

3.3 # Words

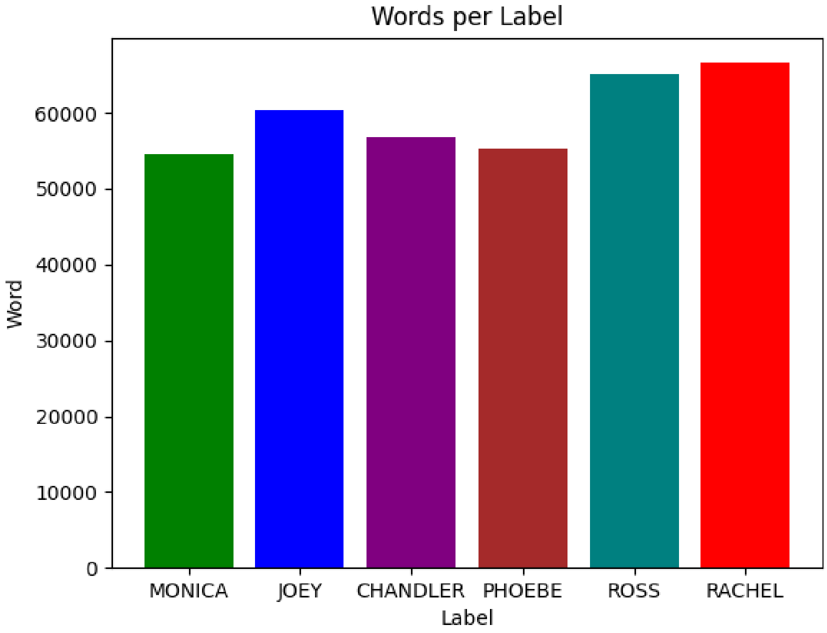

Figure 3: # Words per Label(Character)

	Monica	Joey	Chandler	Phoebe	Ross	Rachel
#Word	54501	60444	56851	55224	65085	66605

Table 3: # Words per Label(Character)

3.4 # Distinct Words

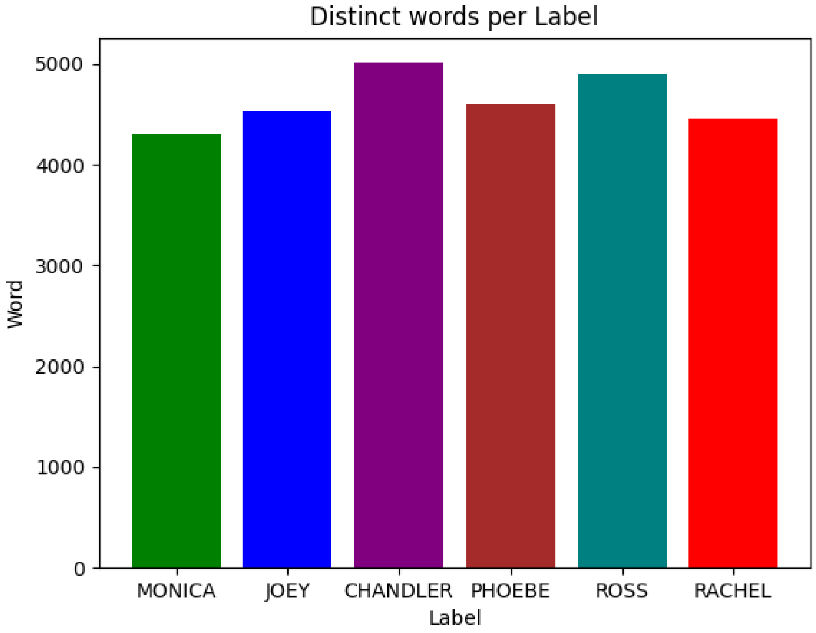

Figure 4: # Distinct Words per Label(Character)

	Monica	Joey	Chandler	Phoebe	Ross	Rachel
#Word	4295	4531	5009	4598	4897	4456

Table 4: # Distinct Words per Label(Character)

3.5 # Common Distinct Words

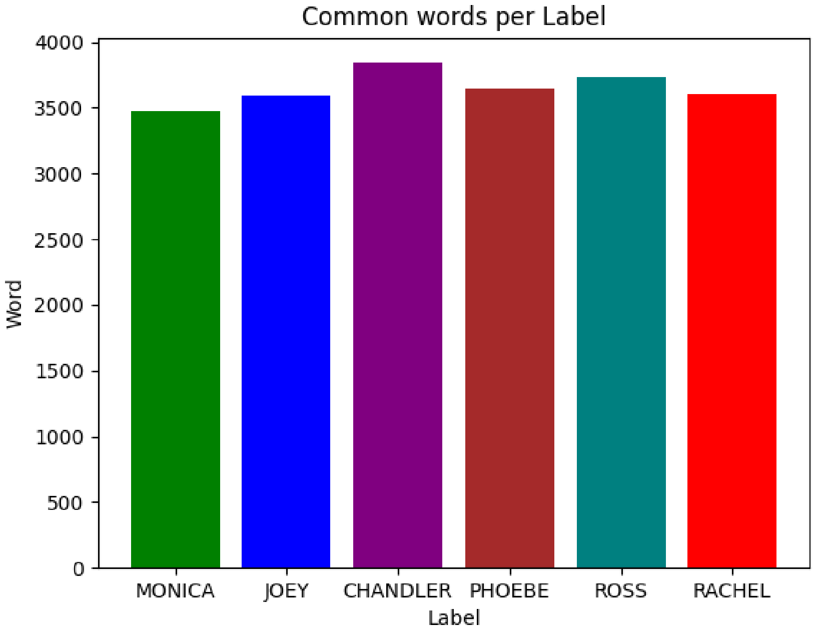

Figure 5: # Common Distinct Words per Label(Character)

	Monica	Joey	Chandler	Phoebe	Ross	Rachel
#Word	3474	3592	3841	3643	3739	3607

Table 5: # Common Distinct Words per Label(Character)

3.6 # Uncommon Distinct Words

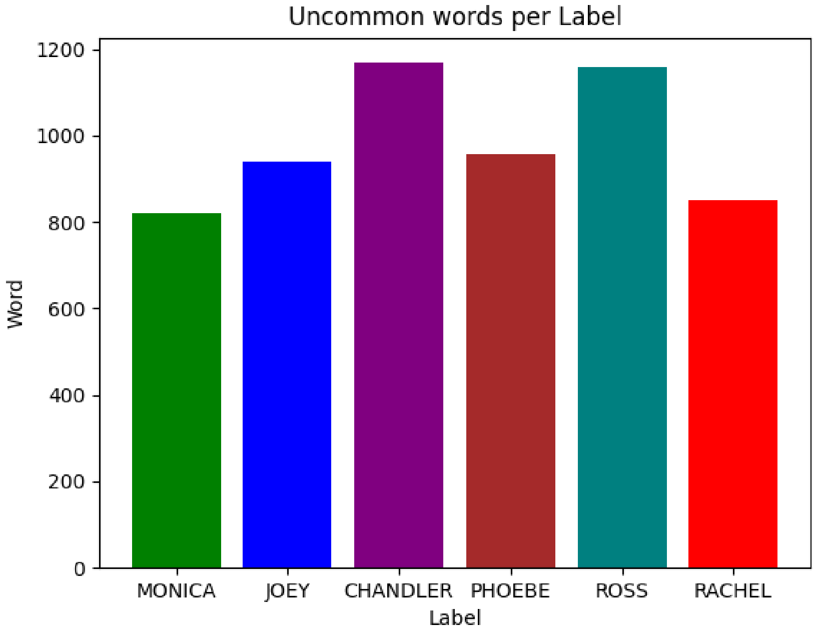

Figure 6: # Uncommon Distinct Words per Label(Character)

	Monica	Joey	Chandler	Phoebe	Ross	Rachel
#Word	821	939	1168	955	1158	849

Table 6: # Uncommon Distinct Words per Label(Character)

3.7 Top-10, By Uncommon Frequency

3.7.1 Label 0: Monica

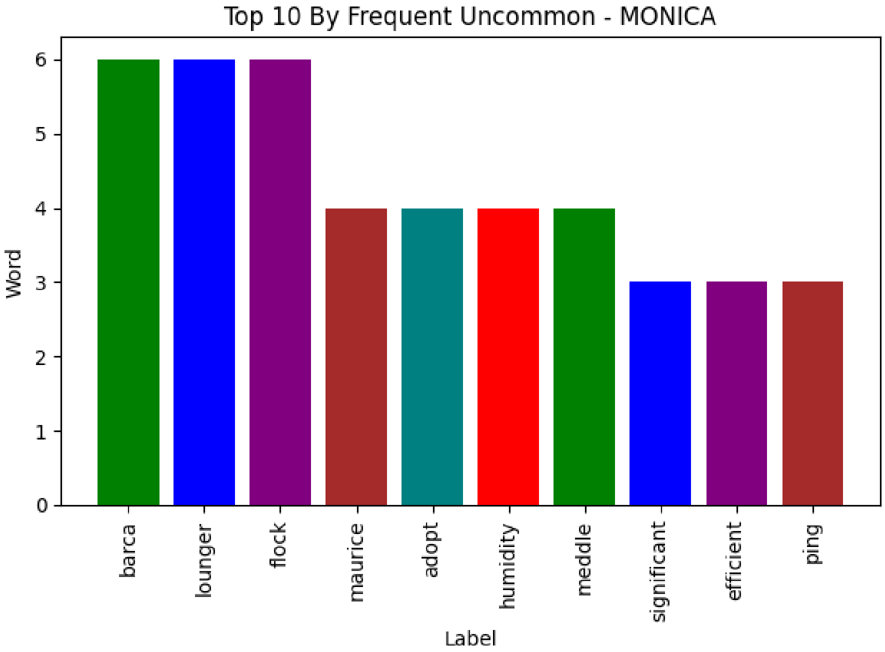

Figure 7: Top10, Frequency, Monica

barca	lounger	flock	maurice	adopt	humidity	meddle	significant	efficient	ping
6	6	6	4	4	4	4	3	3	3

Table 7: Top10, Frequency, Monica

3.7.2 Label 1: Joey

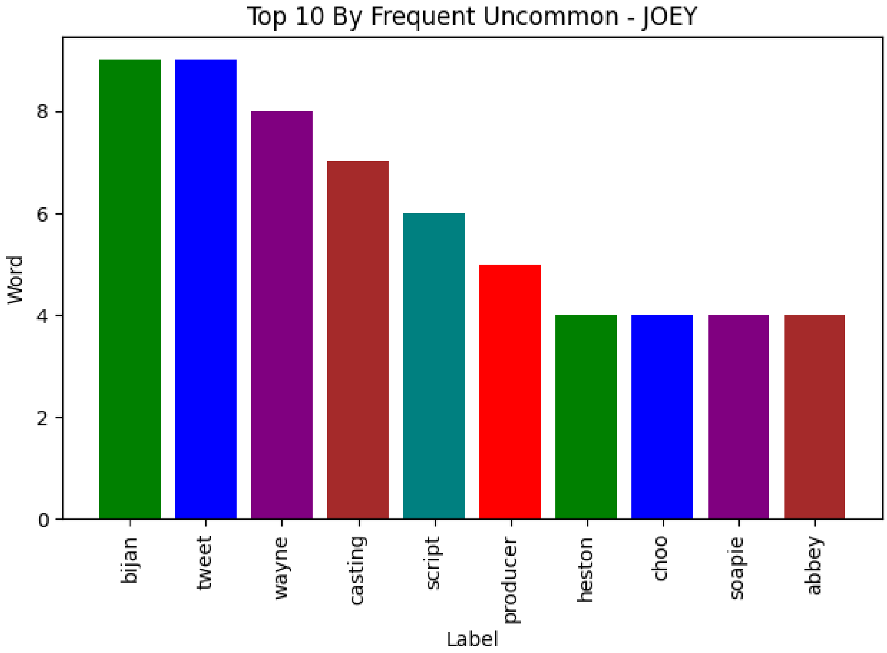

Figure 8: Top10, Frequency, Joey

bijan	tweet	wayne	casting	script	producer	heston	choo	soapie	abbey
9	9	8	7	6	5	4	4	4	4

Table 8: Top10, Frequency, Joey

3.7.3 Label 2: Chandler

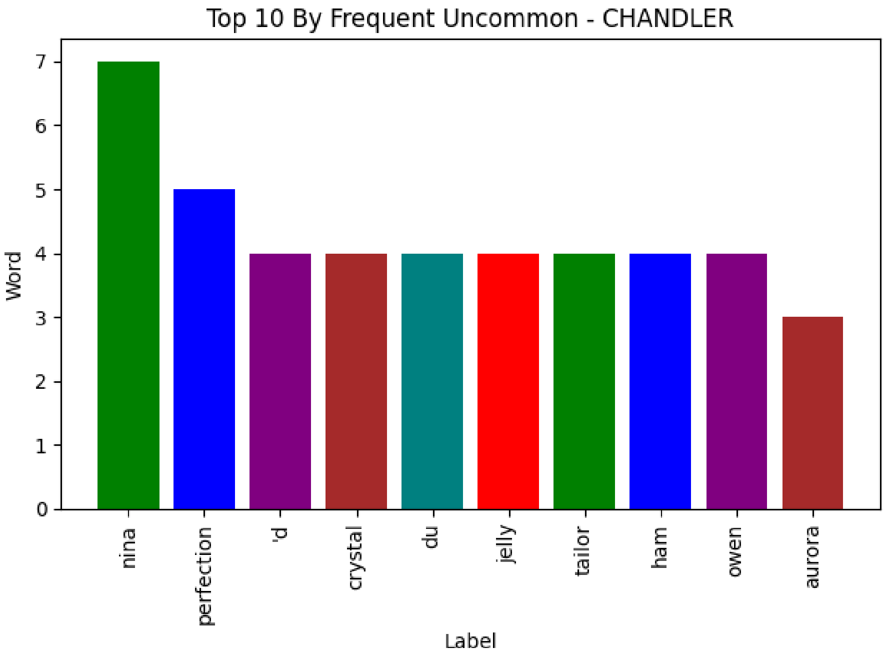

Figure 9: Top10, Frequency, Chandler

nina	perfection	’d	crystal	du	jelly	tailor	ham	owen	aurora
7	5	4	4	4	4	4	4	4	3

Table 9: Top10, Frequency, Chandler

3.7.4 Label 3: Phoebe

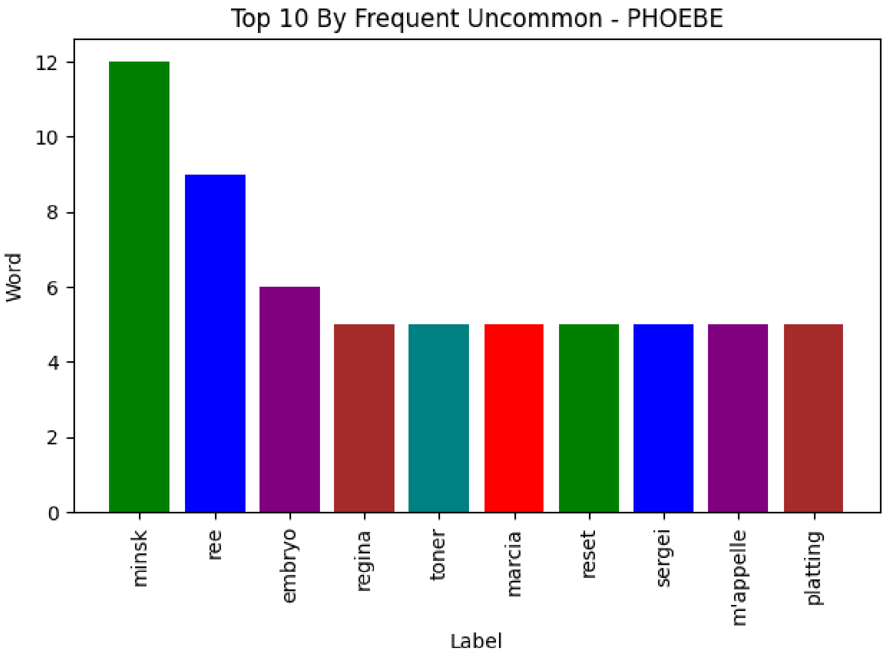

Figure 10: Top10, Frequency, Phoebe

minsk	ree	embryo	regina	toner	marcia	reset	sergei	m’appelle	platting
12	9	6	5	5	5	5	5	5	5

Table 10: Top10, Frequency, Phoebe

3.7.5 Label 4: Ross

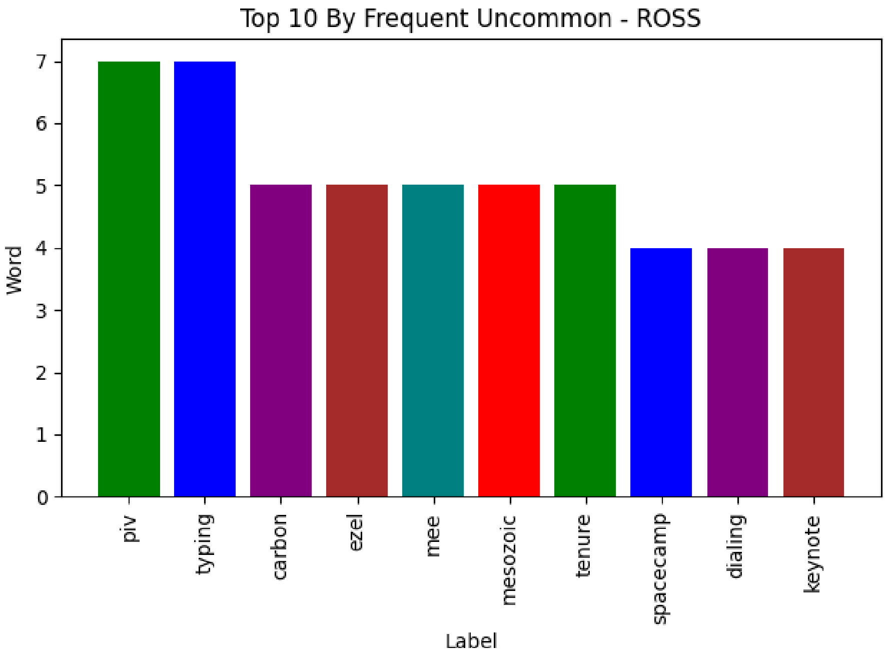

Figure 11: Top10, Frequency, Ross

piv	typing	carbon	ezel	mee	mesozoic	tenure	spacecamp	dialing	keynote
7	7	5	5	5	5	5	4	4	4

Table 11: Top10, Frequency, Ross

3.7.6 Label 5: Rachel

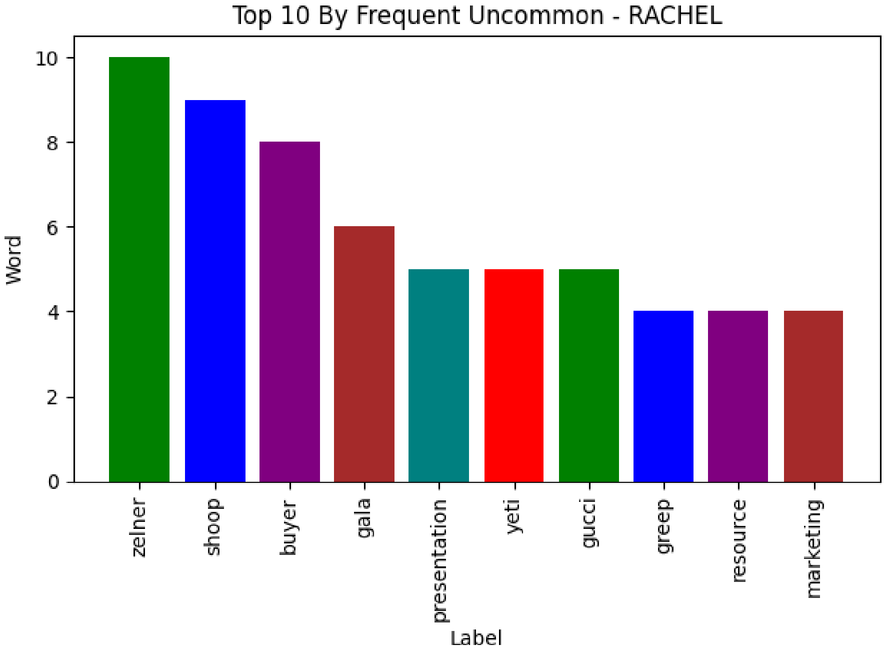

Figure 12: Top10, Frequency, Rachel

zelner	shoop	buyer	gala	presentation	yeti	gucci	greep	resource	marketing
10	9	8	6	5	5	5	4	4	4

Table 12: Top10, Frequency, Rachel

3.8 Top-10, By RNF

3.8.1 Label 0: Monica

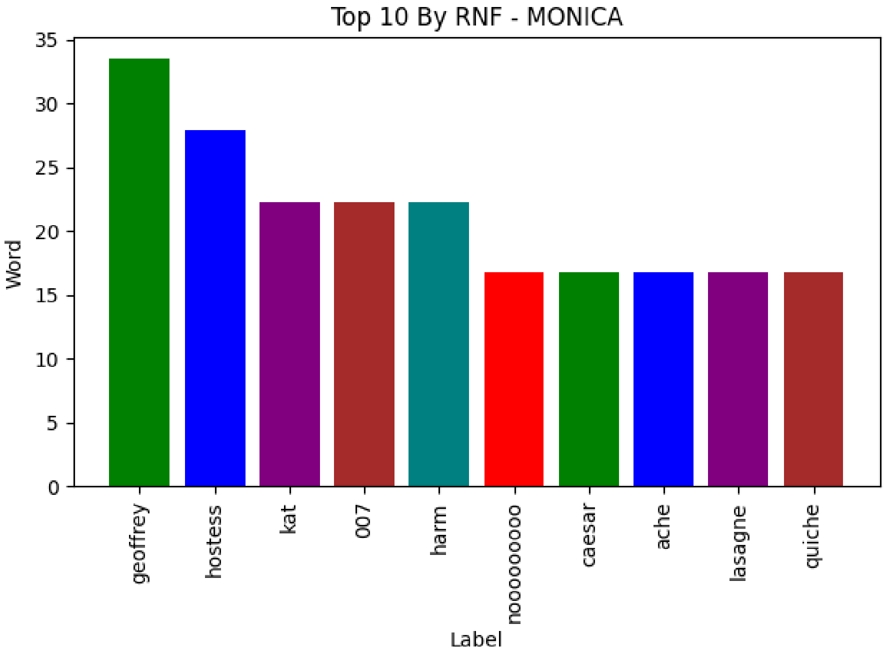

Figure 13: Top10, RNF, Monica

geoffrey	hostess	kat	harm	007	limited	organize	faked	ache	lasagne
33.49	27.91	22.33	22.33	22.33	16.75	16.75	16.75	16.75	16.75

Table 13: Top10, RNF, Monica

3.8.2 Label 1: Joey

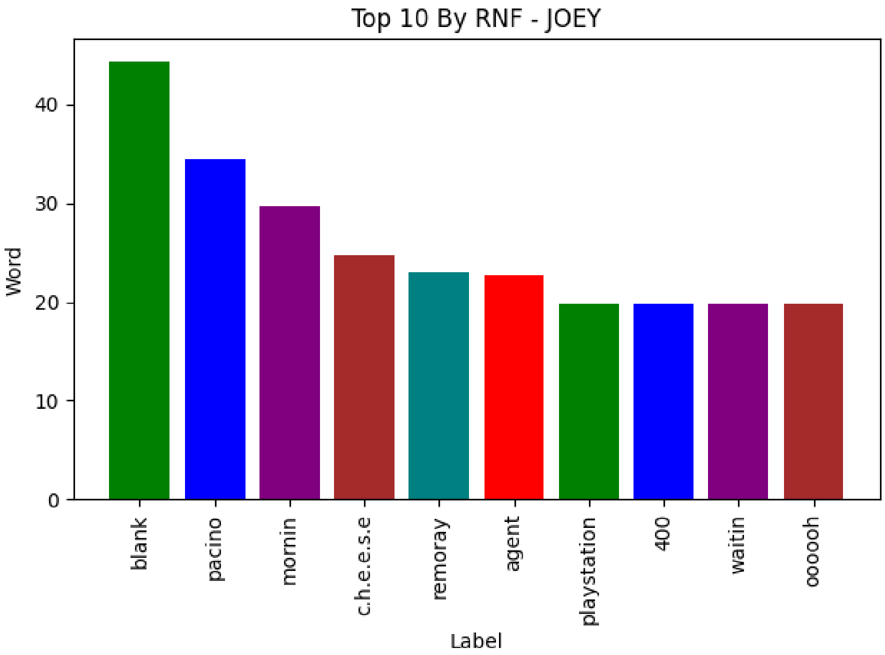

Figure 14: Top10, RNF, Joey

blank	pacino	mornin	c.h.e.e.s.e	remoray	agent	oooooh	vicar	waitin	playstation
44.41	34.54	29.61	24.67	23.03	22.7	19.74	19.74	19.74	19.74

Table 14: Top10, RNF, Joey

3.8.3 Label 2: Chandler

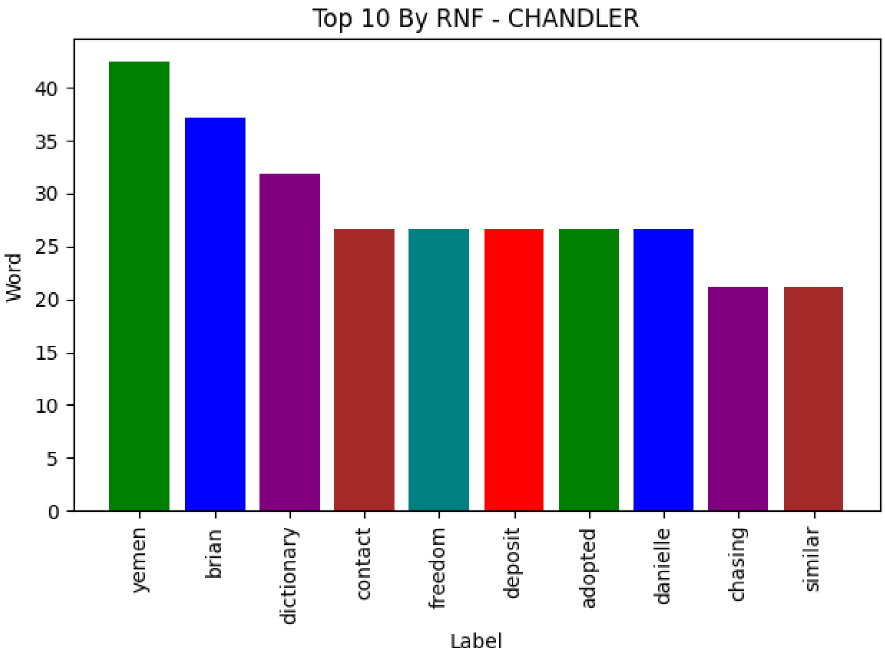

Figure 15: Top10, RNF, Chandler

yemen	brian	dictionary	contact	danielle	freedom	deposit	adopted	chasing	carton
42.48	37.17	31.86	26.55	26.55	26.55	26.55	26.55	21.24	21.24

Table 15: Top10, RNF, Chandler

3.8.4 Label 3: Phoebe

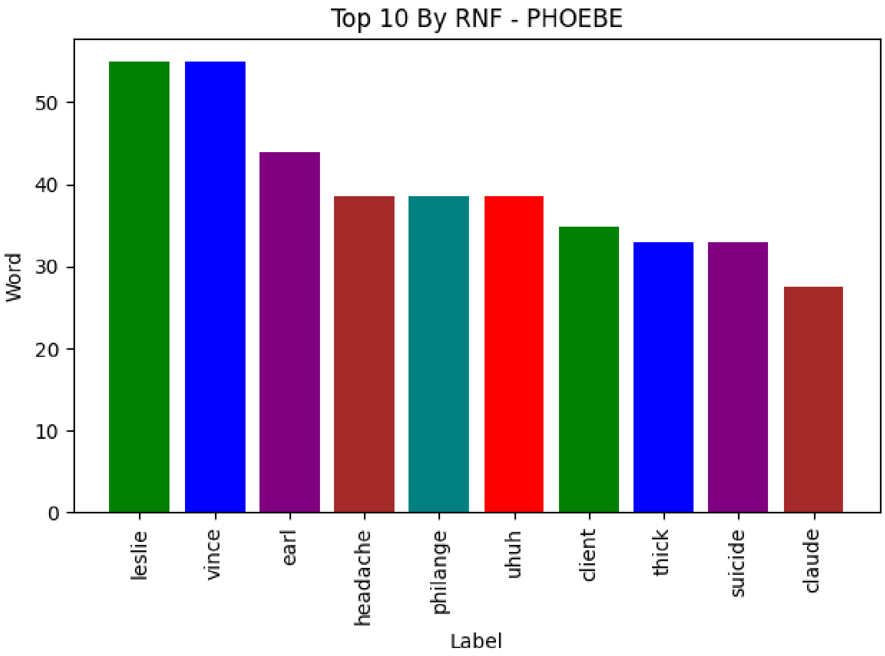

Figure 16: Top10, RNF, Phoebe

vince	leslie	earl	uhuh	headache	philange	client	thick	suicide	claude
54.96	54.96	43.96	38.47	38.47	38.47	34.81	32.97	32.97	27.48

Table 16: Top10, RNF, Phoebe

3.8.5 Label 4: Ross

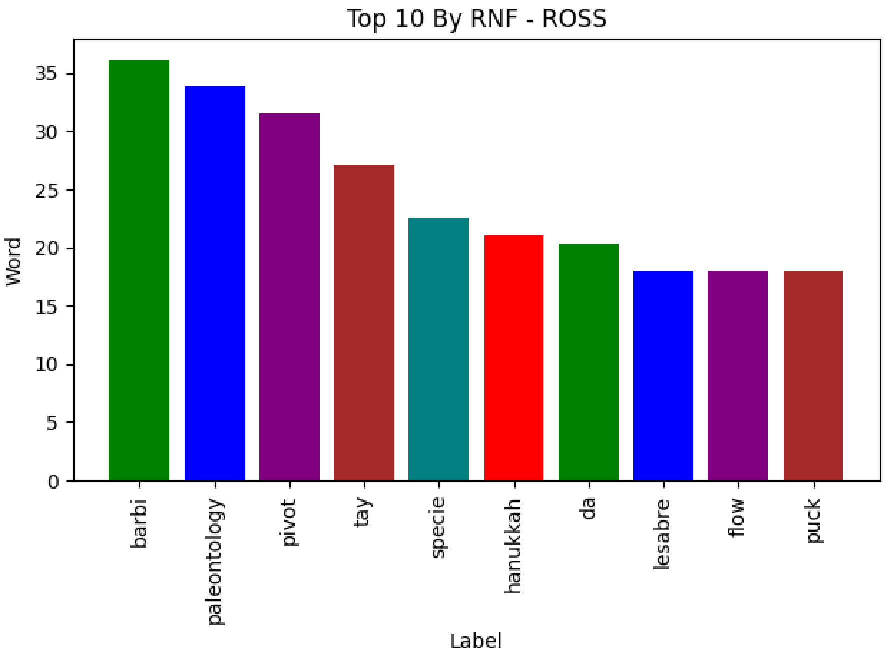

Figure 17: Top10, RNF, Ross

barbi	paleontology	pivot	tay	specie	hanukkah	da	ferry	lesabre	ancient
36.09	33.84	31.58	27.07	22.56	21.05	20.3	18.05	18.05	18.05

Table 17: Top10, RNF, Ross

3.8.6 Label 5: Rachel

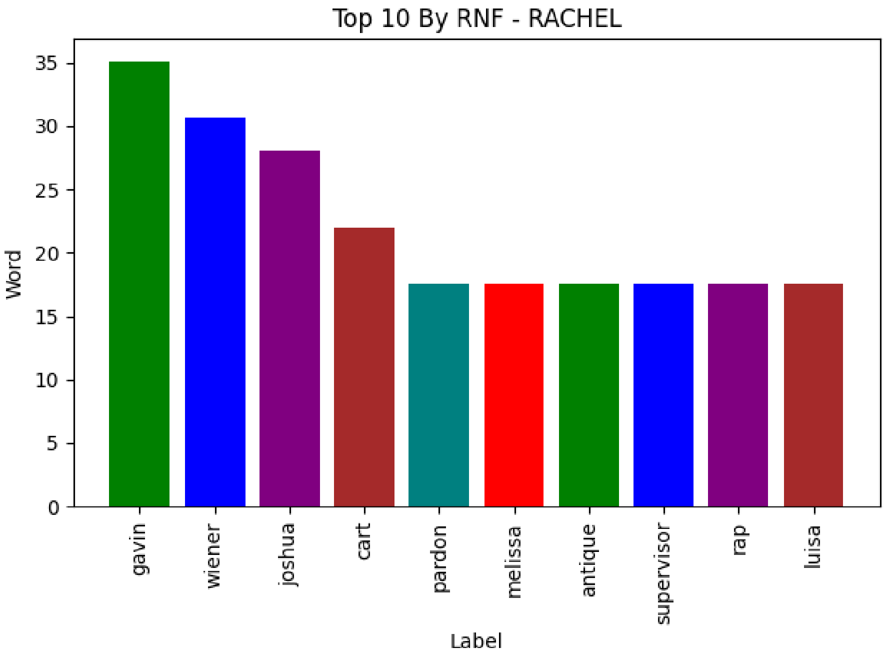

Figure 18: Top10, RNF, Rachel

gavin	wiener	joshua	cart	melissa	rap	mum	luisa	antique	contract
35.09	30.7	28.07	21.93	17.54	17.54	17.54	17.54	17.54	17.54

Table 18: Top10, RNF, Rachel

3.9 Top-10, By TF-IDF

3.9.1 Label 0: Monica

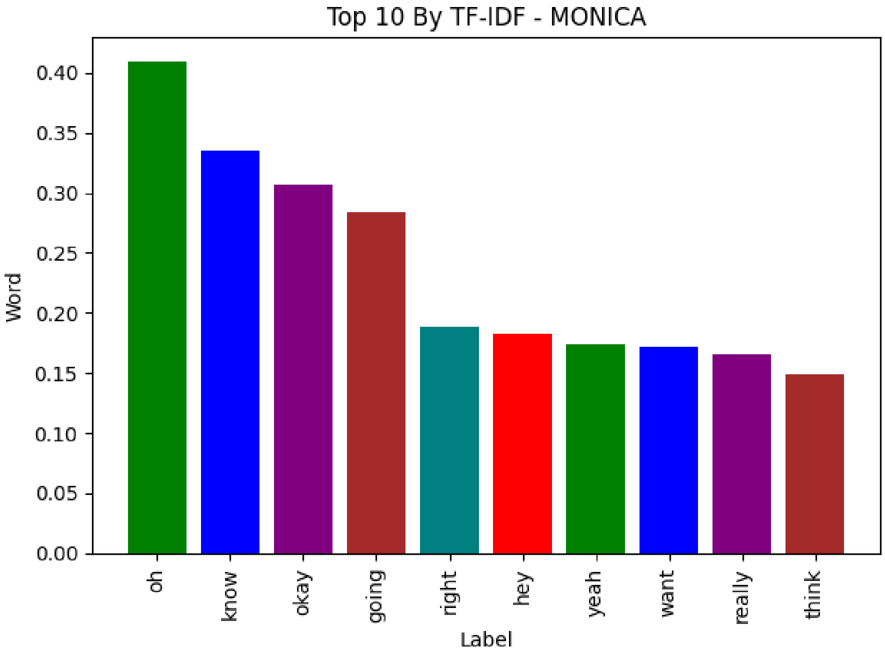

Figure 19: Top10, TF-IDF, Monica

oh	know	okay	going	right	hey	yeah	want	really	think
0.41	0.34	0.31	0.28	0.19	0.18	0.17	0.17	0.17	0.15

Table 19: Top10, TF-IDF, Monica

3.9.2 Label 1: Joey

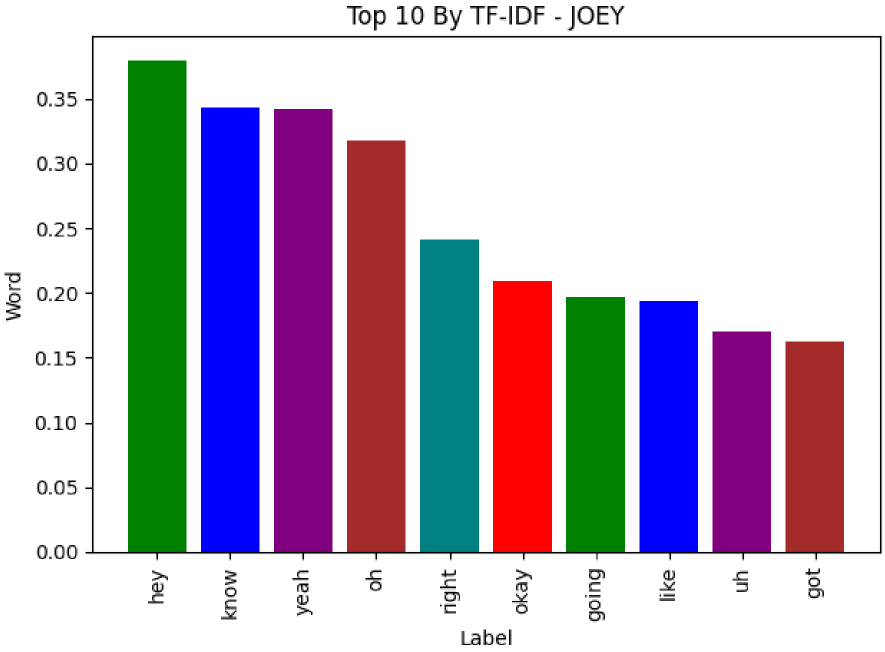

Figure 20: Top10, TF-IDF, Joey

hey	know	yeah	oh	right	okay	going	like	uh	got
0.38	0.34	0.34	0.32	0.24	0.21	0.2	0.19	0.17	0.16

Table 20: Top10, TF-IDF, Joey

3.9.3 Label 2: Chandler

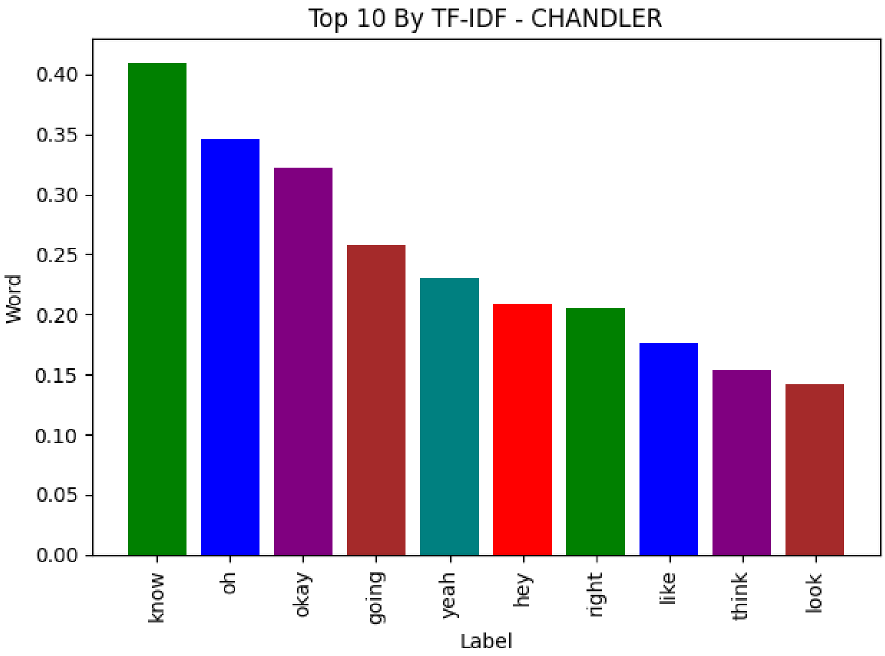

Figure 21: Top10, TF-IDF, Chandler

know	oh	okay	going	yeah	hey	right	like	think	look
0.41	0.35	0.32	0.26	0.23	0.21	0.2	0.18	0.15	0.14

Table 21: Top10, TF-IDF, Chandler

3.9.4 Label 3: Phoebe

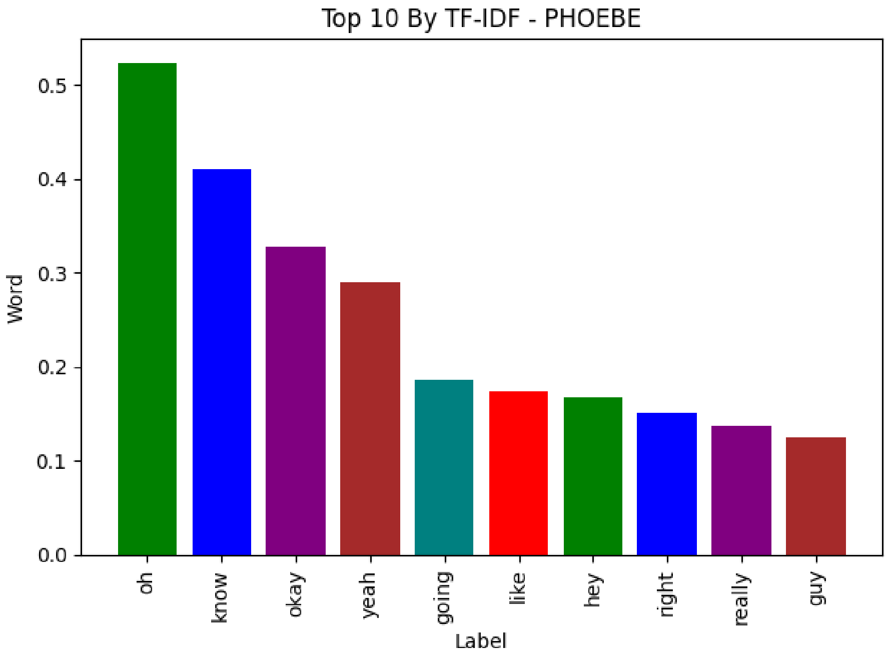

Figure 22: Top10, TF-IDF, Phoebe

oh	know	okay	yeah	going	like	hey	right	really	guy
0.52	0.41	0.33	0.29	0.19	0.17	0.17	0.15	0.14	0.12

Table 22: Top10, TF-IDF, Phoebe

3.9.5 Label 4: Ross

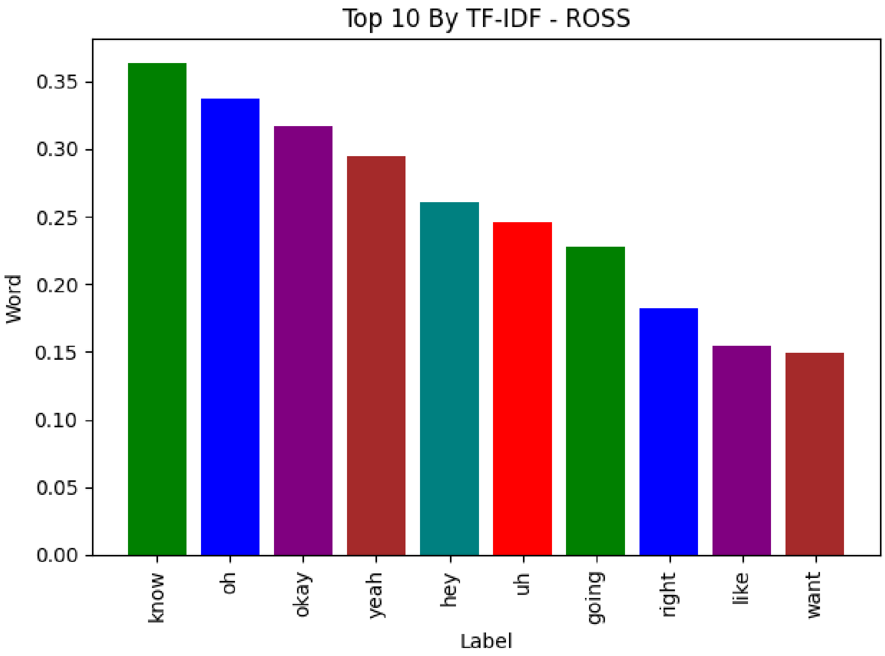

Figure 23: Top10, TF-IDF, Ross

know	oh	okay	yeah	hey	uh	going	right	like	want
0.36	0.34	0.32	0.29	0.26	0.25	0.23	0.18	0.15	0.15

Table 23: Top10, TF-IDF, Ross

3.9.6 Label 5: Rachel

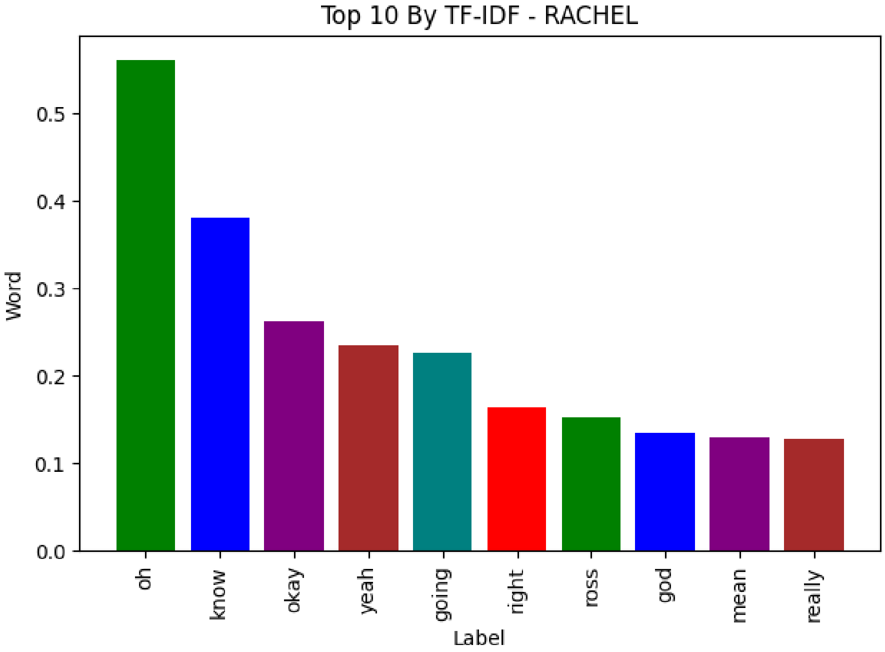

Figure 24: Top10, TF-IDF, Rachel

oh	know	okay	yeah	going	right	ross	god	mean	really
0.56	0.38	0.26	0.23	0.23	0.17	0.15	0.13	0.13	0.13

Table 24: Top10, TF-IDF, Rachel

3.10 Histogram

Histogram files are too vague, and big. Please visit `/analysis/charts/Histogram/` directory for histogram plots for each label.