CROSS-CULTURAL DIFFERENCES IN NEWS: USER INTERFACE EXPANSION

Shaun Wang (sw3048)
Supervisor: Dr. John R. Kender
Columbia University

1. INTRODUCTION

Over the last decade, globalization has exacerbated the availability of news as both media production and consumption have steered towards digitization and social platforms. Coverage of single events across multiple sources and diverse mediums has created a de-territorialized news ecology across the globe, where news coverage of the same event differs in their portrayals across differentiating affinity groups.

In this report, we intended to further expand upon the user interface that displays information through a timeline of the similarities within our project, "*Tagging and Browsing Videos According to the Preferences of Differing Affinity Groups.*" Our goal was to add additional features and components to the end user experience and provide them with additional information derived from our ML algorithms.

2. PREVIOUS STUDIES

From their previous work, '*Displaying Cross-Cultural Differences in News Videos III'* Luvena Huo and Tiansheng Sun [3] perfected the user interface design created by David Dirnfield [1] and Jiaqi Liu [2], which included connecting the similarities of two affinity groups of the same news coverage on a similarities timeline. Luvena Huo and Tiansheng Sun increased the clarity and usability of the interface through creating a *'Show Commonalities'* button while also including a *Finalized Information Display* for commonality pairs. The design was simple-to-use and clearly exemplified the timeline's functionality. This semester, we wanted to further improve the components of the user interface in order to deliver more information when it comes to analyzing differentiating news coverage on the same event, such as differences amongst the news coverage.

2.1 Issues with Existing Display of Information

Currently, our existing wireframe includes: a landing page, a dropdown list to select the event, a timeline that compares the events, and a show commonality button.

Fig 1. Existing Wireframe.

Since our goal with this tool is to be public facing, this means that we need to provide context for the end-user. As an end-user, the

tool needs to provide a context/summary on the event and affinity groups that are being compared in order to situate the audience. Currently, the only indication that these two timeline similarities are being compared is through its *Information Pop-Up Modal* and the *Show Similarities Button*.
It needs a title to clarify that only the similarities are displayed, and the event-comparison page needs to summarize the event that is being compared. Since we already have algorithms to extract key-frames from videos and they are being compared as similarities, we can also include a 'most-frequent key-frame' to situate the audience. Moreover, another side of the comparison that we are missing is the differences amongst the comparison.

2.2 Expanding the User Interface into Difference Comparison

With that in mind, we wanted to explore and research a visual component that could present the differences across the two affinity group's new portrayal. The question becomes, what are we basing our comparison on? What is the quantitative metric that can be used for difference comparison? And what is the best visual format to present differences?

3. RESEARCH ON ADDITIONAL DISPLAY COMPONENTS

After extensive research on the most optimal visual format to display differences, we came up with three different visual possibilities to compare the differences by using key frames or tags generated from our algorithms:

1. Tags Comparisons through Word Cloud Generation
2. Dissimilarity within Similar Key Frames or Tags
3. Comparison through Lists, Specs, or Features, and Side-by-Side Infographics

3.1 Tags Comparisons through Word Cloud Generation

We found a word cloud that was generated from a social structural topic modeling study. Taken from *"This a Liberal and Conservative Representations of the Good Society: A (Social) Structural Topic Modeling Approach"*[4], the study used 'quantitative text-analytic methods to analyze more than 3.8 million messages sent by over 1 million Twitter users about what constitutes a good (vs. bad) society.' We saw that the word-cloud was clearly able to present a coherent comparison across two different affinity groups. The word-cloud included keywords that suggest *'that in writing about the "bad society," where liberals were more likely than conservatives*

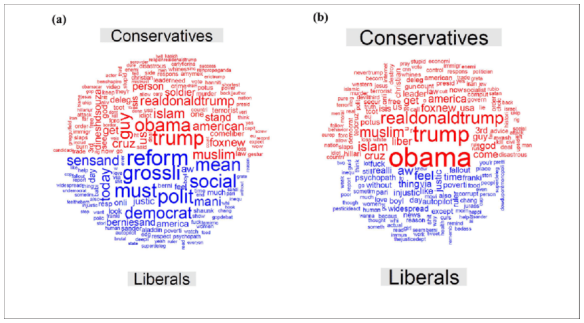


Fig 2. Comparison word clouds displaying ideologically divergent representations of the "bad society": (a) includes all tweets collected and (b) excludes retweets. [4]

to use words and phrases such as inequality, justice, injustice, poor, poverty, food (insecurity).' While *'Conservatives, on the other hand, were more likely than liberals to use words and phrases such as Muslim, Islam(ic), terror(ist), ISIS, moral, order, control, religion, pray, Jesus...'* More importantly, we thought the utilization of a word-cloud as a juxtaposition would be beneficial to comparing portrayals of differentiating affinity groups.

3.2 Dissimilarity within Similar Key Frames or Tags

Another way to compare dissimilarity is through anchoring with either key-frames or tags and then comparing what is different surrounding that anchor or timeframe. For instance, if a key-frame at minute 3:00 is the same and the tags are different, then we would pinpoint this location and present what the differences are.

Fig 3. A video of Nancy, where one video was altered.

We researched this New York Times article. The NYT presents the Distorted Videos of Nancy Pelosi Case Study, where on the left includes an original video of Pelosi, while the right shows an altered video of her, where she is stuttering. Moreover within this video, while the key frames are similar, the meaning, and thereby tags, generated are different.

3.3 Comparison through Lists, Specs, or Features, and Side-by-Side Infographics

While comparisons through specifications or features is one of the most common ways to differentiate products, this format of comparison presents many issues.
As we derive key-frames and tags, it is hard to pinpoint comparable and quantitative metrics, since different events and affinity groups all have their own range of metrics. Unless we can implement some form of semantic analysis, which unifies tags into specified dimensions.

3.4 Decision on Visual Format

As we conducted more research into word-clouds, we found existing resources and repositories that could easily implement word-clouds. This allowed us to decide word-cloud as our visual format to present differences, as we could effectively input our data through existing components. We also needed word-clouds to present in various languages, where we found additional language libraries with ease.

4. WORD-CLOUDS

Fig 4. A word-cloud presenting the Constitution.

The existing Github repository that we will be using is Word-Cloud by Andreas Mueller. It's dependencies included numpy and pillow.

Link to Github Repository:
[7] https://github.com/amueller/word_cloud
Link to Documentation:
[7] http://amueller.github.io/word_cloud/

4.1 Word Cloud Repository Extension: Chinese Library, Jieba

Since we are comparing differentiating affinity groups, it was crucial to include the capabilities to compare other languages. While Word Cloud in its standalone library does not support Chinese, its documentation displays how to implement Chinese word cloud. To implement Chinese, we will have to use a segmentation library called, Jieba. Jieba is now the most modern and popular Chinese word segmentation tool in python. We can use 'PIP install jieba'. To install it.

Link to Github Repository:
[8] https://github.com/fxsjy/jieba

4.2 Additional Features

The word-cloud repository included other extensions such as Colored by Group, or Image-colored with boundary map. Using a mask or an image-based coloring strategy, we could generate word clouds in arbitrary shapes by utilizing the ImageColorGenerator method. The word-cloud uses the average

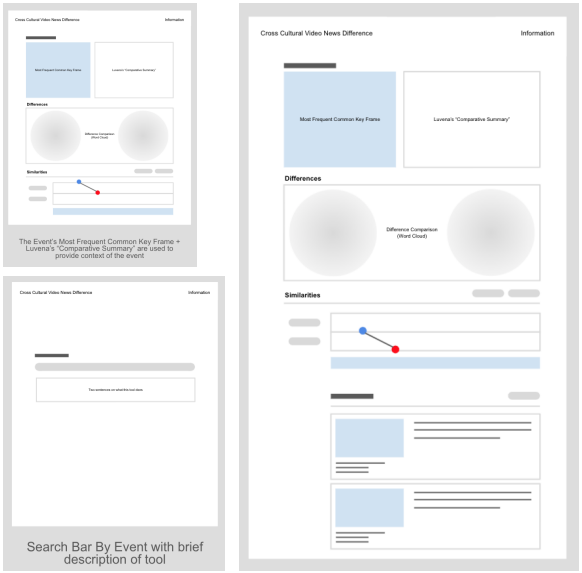
Fig 5. A Image-colored word cloud on Alice [7]

color of the region occupied by the word in a source image to determine its legal color. We could also pass a different image to the mask, which will be interpreted as 'don't occupy' by the WordCloud object. More importantly, this means we could generate a word cloud that assigns colors to words based on a predefined mapping from colors to words. This would allow us to categorize different affinity groups with different colors.

5. PROPOSED WIREFRAME

We drafted up a new wireframe that intends to situate the audience through a common image, a title, and two word-clouds as the visual format to exemplify differences:

Fig 6. Newly Proposed Wireframe.



The newly proposed wireframe includes:

- On the top left blue component, we include a most frequent image amongst the similarity key-frames from both affinity groups.
- To its right, include a NLP driven summary of the event. (Future implementation)
- A title above the word-cloud named: *Differences.*
- Two word-clouds from two affinity groups in juxtaposition.
- A title above the timeline named: *Similarities.*

5.1 Proposed Wireframe Shortcomings

We foresee that there are shortcomings to this wireframe:

- Side-by-side comparisons fail to include 2+ affinity groups. This needs further investigation.
- We need to include middle words, since word-clouds are based upon frequency of the word. .

5.2 Possible Additional Features

Other possible features we could implement in the future include:

- On each event page, we could generate statistics for each event. E.g duration, number of tags / key frames. etc
- News engagement metrics *(require external metrics for this)*
- Comparison to the affinity group's global metrics *(requires more # of compared-events)*

6. IMPLEMENTATION & CODE

6.1 Dependencies Requirements for Word-Cloud

The dependencies that are needed for word-cloud includes the Multidict Module, Numpy, and Pillow.

- Multidict is a dict-like collection of key-value pairs where a key might occur more than once in the container. This data structure is required to feed into the WordCloud() method, since HTTP Headers and URL query string require specific data structure: multidict.
- Pillow is a Python Imaging Library that adds image processing capabilities to word-cloud.

In addition, the word-cloud method parameter only allows for strings with space delimiters or multidict with frequency columns.

6.2 Implementing Real World Data

I partnered with Jimmy Zhang, as he was working on extracting tags in Chinese. He used K-means to generate Chinese tags. We thought that having a word-cloud that also generates tags based upon a frequency algorithm is redundant. Therefore, we agreed that if he could provide a dictionary structure for both Chinese and English, we

could eliminate the frequency algorithm within word-clouds.

The data pipeline needed for word cloud with Jimmy's data output:

- DataFrame ⇒ txt file ⇒ python dictionary ⇒ multi dict
- Currently, this pipeline is based on having separate storage for his K-means algorithm and word-cloud.
- However, if they were all implemented into the same repository, then we could eliminate the txt file.

6.3 CODE

```
import multidict as multidict
import numpy as np
import os
import re
from PIL import Image #use if need mask
from wordcloud import WordCloud
import matplotlib.pyplot as plt

def getFrequencyDictForText(sentence):
    fullTermsDict = multidict.MultiDict()
    tmpDict = {}

    # making dict for counting frequencies | might need to be replaced with Jimmy's code
    for text in sentence.split(" "):
        if re.match("a|the|an|the|to|in|for|of|or|by|with|is|on|that|be", text):
            continue

        if text in tmpDict.keys():
            continue

        val = tmpDict.get(text, 0)
        tmpDict[text.lower()] = val + 1

    for key in tmpDict:
        fullTermsDict.add(key, tmpDict[key])
    return fullTermsDict


def makeImage(text):

    wc = WordCloud(background_color="white", max_words=500)
    # generate word cloud
    wc.generate_from_frequencies(text)

    # show
    plt.imshow(wc, interpolation="bilinear")
    plt.axis("off")
    plt.show()


# get data directory (using getcwd() is needed to support running example in generated IPython notebook)
d = path.dirname(__file__) if "__file__" in locals() else os.getcwd()

text = open(path.join(d, 'Coding/AlphaGo_ENG.txt'), encoding='utf-8')
text = text.read()
makeImage(getFrequencyDictForText(text))
```

7. RESULTS

7.1 Word-Cloud Generated from Real Data

These word-clouds are generated with Jimmy's data. This data is generated through Jimmy's *get_n_level_frequency_words_with_kMeans* method call with 3 levels of k-means:

Fig 8. Word-Clouds w/o color generated with real data.

We have yet to create a k-means algorithm for English tags; therefore, the existing word-cloud for english is generated through a word-frequency algorithm.

7.2 Updated User Interface

We implemented the word-cloud with the specifications from the newly proposed wireframe into a prototype:

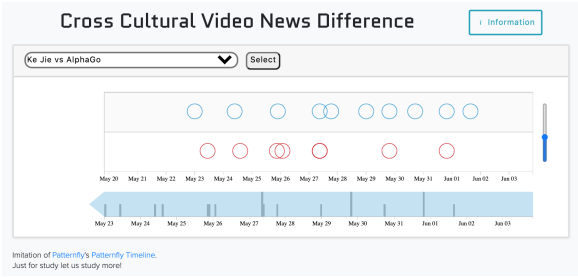


Fig 9. Previous Prototype

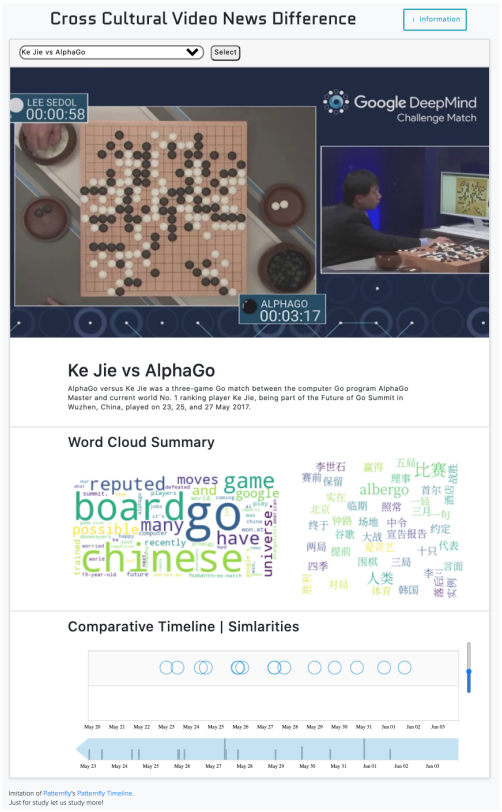


Fig 10. Prototype with updated Wireframe.

In this updated user interface, we included all of the new specifications within the newly proposed wireframe. That said, since most of the project has been dedicated to generating word-clouds and tags, we left the index.html static. In the future, we need to implement an algorithm that would pull all of these resources from a centralized storage.

8. FUTURE WORK & ADDITIONAL RESEARCH

Our existing prototype has many unimplemented features. We have also researched ahead to see what are additional possibilities before we implement any changes to the source code.

8.1 Correlating English with Chinese

In the future, we want to correlate Chinese with English and differentiate similarities and differences amongst the two with colors. For this to happen, we need to first translate the Chinese into english and compare their similarities and differences.

This also raised many questions; for instance, Should we translate into Chinese ⇒ English or English ⇒ Chinese, or both? Which language do we use as a benchmark for translation, since even when definitions are similar to the viewer, the meaning is altered after translation? We also discussed how we could use color to show cultural uniqueness in common chinese / english and associate color with similarity and differences (e.g red with difference, blue for similarity) More importantly, we will need to correlate the two languages into a textual similarity algorithm.

Since there will be differences within the translation, the text similarity algorithm has to determine how 'close' two pieces of text are both in lexical similarity and semantic similarity. We hope that by comparing the two texts, generating its accuracy in correspondence. We can use the accuracy to evaluate the color in the word-cloud in the future.

8.2 Proposed Future Word-Cloud Prototype

With the suggestion from Prof. Kender, we also explored if there are bilingual word-clouds. Or, could we have word-clouds in different languages stacked onto each other, instead of two seperate world-clouds.

Derived from Prof.Kender's drawings:

Taking sometime to research, we thought Stacked Word Cloud is similar to a union, so that we fit them together:

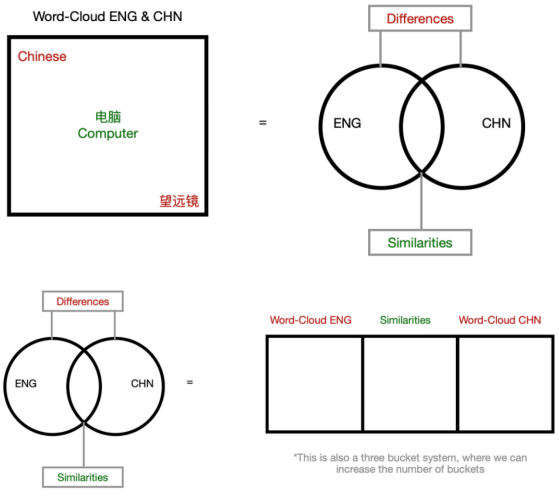


Fig 11. 'Stacked' Design for Word-Cloud Component

We can achieve the illusion of a stacked word cloud through combining three word-clouds, without altering or refactoring too much of word-cloud source code. Since it is extremely difficult to include two types of language into one word cloud, this design allows us to have the union of the two languages in one, while also keeping the differences to the other two word-clouds. We can also increase the number of word-clouds in the middle to create a bucket/gradient system of similarities and differences. Moreover, to achieve this we will need to:

- Semantic analysis of different languages to create a correlation between the two languages.
- An algorithm that codifies frequencies and color correlation.

8.3 Additional Future Work

To fully make our tool fully functional, we need to further dive into front-end development in the future. This includes:

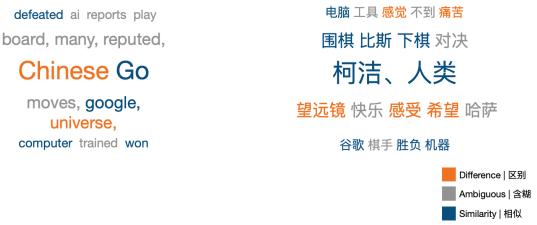
- Refactoring existing code into components.
- Developing full frontend using frameworks such as React, Angular, etc.
- Implementing existing static components.

8.4 Color Theory for Similarity / Differences

We also conducted testing on text color for differences and similarities through surveying. Our goal was to see if Chinese vs English affects color preference based upon textual language and audience's nationality.

- We drew a template, since there isn't an algorithm to match Chinese meaning with that of English yet.
- Messaged friends with color variation A & B, asked which one makes it more clear that which better colors are similar and others are different.
- The testing included a sample of 15 friends with 2 no responses. (Mixture of American friends and Chinese friends)

Color *Variation A*



Color Variation B

Fig 11. Testing Color preference through Google Survey. Result: 89% Color Variation A (top). 31% Color Variation B (bottom).

9. PROTOTYPE & GITHUB

Link to the current GitHub: https://github.com/shaunwang1350/CrossCulturalMediaAnalysis_WordCloud

Currently, the newly implemented wireframe is static, which requires further implementation in the future. To navigate

the current prototype, first git clone the repository to your existing device. Then download the required libraries. Navigate to the correct folder with your command line and type:

```
1    explort Flask_APP = app.py
2    flask fun
```

10. REFERENCES

[1] Jiaqi Liu. Displaying Cross Cultural Differences in News Videos. http://www.cs.columbia.edu/~jrk/NSFgrants/video affinity/Interim/19x jiaqi.pdf

[2] David Dirnfeld. Displaying Cross Cultural Differences in News Videos II. http://www.cs.columbia.edu/~jrk/NSFgrants/video affinity/Interim/20y_dave.pdf

[3] Luvena Huo, Tiansheng Sun. Displaying Cross Cultural Differences in News Videos III. http://www.cs.columbia.edu/~jrk/NSFgrants/videoaffinity/Interim/20x_Luvena_Tiansheng.pdf

[4] Sterling, Joanna & Jost, John & Hardin, Curtis. (2019). Liberal and Conservative Representations of the Good Society: A (Social) Structural Topic Modeling Approach. SAGE Open. 9. 215824401984621. 10.1177/2158244019846211.

[5] Mervosh, Sarah. "Distorted Videos of Nancy Pelosi Spread on Facebook and Twitter, Helped by Trump." *The New York Times*, The New York Times, 24 May 2019, www.nytimes.com/2019/05/24/us/politics/pelosi-doctored-video.html.

[6] Sieg, Adrien. "Text Similarities : Estimate the Degree of Similarity between Two Texts." Medium, Medium, 13 Nov. 2019, medium.com/@adriensieg/text-similarities-da019229c894.

[7] Mueller, Andreas. Word-Cloud. MIT License. https://github.com/amueller/word_cloud

[8] Sun Junyi. Jieba 结巴中文分词. MIT License. https://github.com/fxsjy/jieba