Structured TV Shows — “You Have Been Chopped”

1. Introduction

With an increasing demand for rich, immersive and interactive experiences, viewers want to engage with their favorite TV shows to learn and discuss the characters, backstories and potential arcs in the storyline. Many shows have long story-line arcs spanning multiple episodes and even seasons leading to complex narratives such as “Game of Thrones”. On the other end of the spectrum, we have highly-structured shows which have a tightly scripted format which does not vary much across the episodes and even multiple seasons — examples include the TV shows “Shark Tank”, “Chopped”, “The Voice”, and “American Ninja Warrior” involving entrepreneurs, chefs, singers, and athletes respectively. The structure arises due to the shows using the same venues, music themes for specific transitions and specific words for contextual changes. Chaptering is critical to obtain the contextual information.

Most existing literature focus on video segmentation or chaptering using audio-visual analysis (Sargent et al., 2014; Aoki, 2006). They are not robust to generate semantic chapters. Previous work (Yamamoto et al., 2014) presents a semantic segmentation of TV programs based on detection of corner-subtitles, but the corresponding semantic information is limited to Japanese TV programs.

To build a context-sensitive video question answering system, we leverage the structure of the show to extract metadata by Closed Caption (CC) analysis and visual analysis. *In particular, we incorporate semantics through domain knowledge, in the form of transition phrases, in order to chapter structured TV shows.* The metadata are then used for indexing and retrieval of entities of interest with respect to chapters, and the corresponding chopped segments containing the entities are presented to end users. We build a video question answering system for the TV show “Chopped”, and the system is illustrated in Figure 1.

2. Semantic Analysis of Videos

Data Preparation. We processed 60 videos of the “Chopped” consisting of various episodes in 6 seasons - $(27, 23, 22, 21, 15, 14)$. CC segments were extracted from the raw videos using the `ffmpeg` package. We use information about the show such as cooking ingredients, episode title, names of the judges, participating contestants and the winners of the various rounds extracted from Wikipedia. We extract food-related phrases from the CC using Stanford Core NLP (Manning et al., 2014) and Word2Vec (Mikolov et al., 2013).

Chopped segments. We segment the videos by chapters with very little domain knowledge but leveraging the shared structured across Chopped episodes. First piece of domain knowledge is that each episode consists of three chapters, namely the cooking of appetizers, entrees, and desserts. Also, we obtain seed examples of what the host says during a chapter transition; *e.g.*, when host says “Please open your baskets”, the ingredient basket for the upcoming chapter is revealed – this is used as a signal for the start of the chapter. Similarly, “Chef X, you have been chopped” is a phrase uttered at the end of a chapter. Only with this domain knowledge, we create a small set of regular expressions to catch the CC text signaling a chapter transition.

For i^{th} episode, we find all captions that match the regular expression, which produces a set of candidate start and end markers, S^i and E^i, respectively. While the regular expressions are expected to accurately find the start and end markers in most episodes, typos and linguistic variations might create noise (e.g., the second start marker is between the first start and end markers). Due to this, getting the actual interval for each chapter is not always trivial. To select the optimal chapter intervals, we rely on *reference episodes*, where we have the expected sequence of start-end markers for each chapter (i.e., for ‘Chopped’, this means that we have three start and three end markers, for appetizer, entree and dessert chapters). We find the earliest start time and latest end time for each chapter, across all episodes, denoted by (S_{app},E_{app}); (S_{entr},E_{entr}); (S_{dsrt},E_{dsrt}). Given these reference intervals, as well candidate start-end markers of the i^{th} episode, we compute the optimal start time of each chapter by finding the earliest start marker that falls into the reference interval for that chapter. Similarly, we compute the optimal end time by finding the latest such end marker:

$$S^i_{app} = \min\{s | s \in S^i \wedge s \in (S_{app}, E_{app})\}$$
$$E^i_{app} = \max\{e | e \in E^i \wedge e \in (S_{app}, E_{app})\}$$

OCR on Selected Scenes. Since we know where the contents of the basket are revealed and the ingredients are not included in the closed captions, we recognize the characters on screen by an optical character recognition (OCR) algorithm (Smith, 2007), named Tesseract OCR, on that moment. Specifically, we run the OCR only on possible text region candidates for computational efficiency. The region candidates are found by combining the morphological filtering and the binarization of the images with the Otsu threshold (Otsu, 1979). The Tesseract OCR fine-tunes the word regions in the candidate regions by line finding, baseline fitting and proportional word finding. Then, it encodes each character into a segment feature of polygonal approximation and efficiently classifies it by quantization of the feature vector and map it into a hash code. Once we have the OCR outputs for multiple frames of the food basket ingredients, we clean up and find the food phrases by (a) removing strings that appear in fewer than ten frames, and (b) keeping only phrases that are in the vocabulary of Google word embeddings.

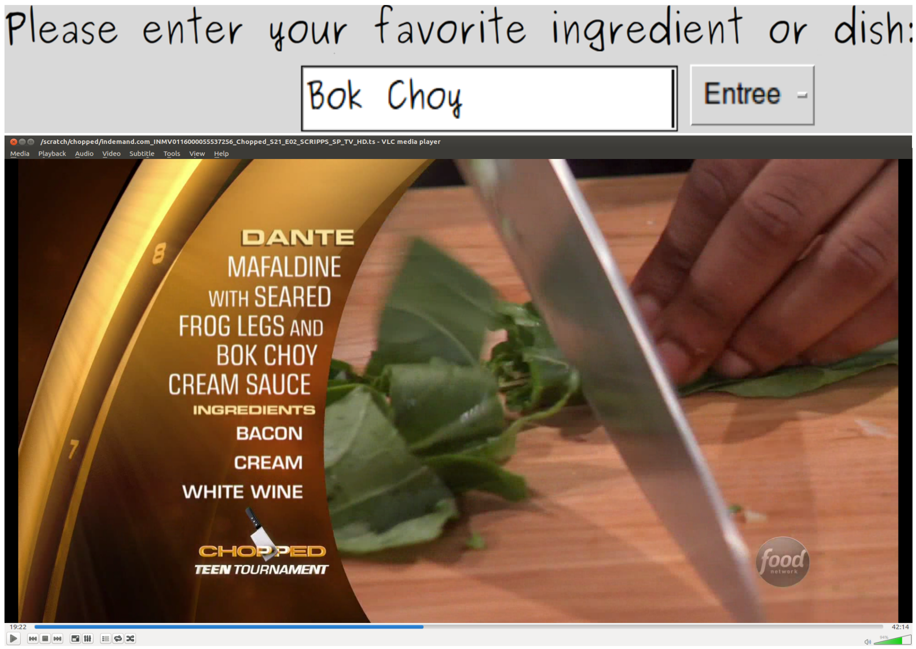

Figure 1. Our system screenshot for input query “bok choy” and retrieved video compilation.

Indexing Food Mentions. Noun phrases that appear in closed captions across all episodes are found using Stanford CoreNLP. We create an inverted index, so that given a noun phrase, we can access all time intervals of each episode where it was mentioned. We also create a filter specifically for Chopped, where we want to distinguish food phrases from other noun phrases. In order to automate this, we take advantage of pre-trained word embeddings (Mikolov et al., 2013). The vector of a noun phrase is the average of the vectors of each word (ignoring stop words). We then compare this to the vector of “food”, and keep phrases with a cosine similarity beyond a threshold (e.g. 0.20 in this case.)

Search Interface. As a use case of the metadata that we extracted from Chopped, we built an application where the user can query an ingredient or dish, from which we generate a custom “Chopped” compilation video. Our approach uses the inverted index of food phrases to retrieve all time intervals in which the query was mentioned, across all episodes. Since each time interval is very short (few seconds on average), we try to merge nearby mentions as much as possible, so that the compiled video is not chopped into too many pieces. For this, for each episode, we sort all retrieved intervals, and merge consecutive ones if the difference is less than 5 seconds — we keep doing this until there are no intervals closer than 5 seconds apart. In order to add more context, we then expand each interval by 3 seconds to the left and right. While this heuristic works fine in many cases, selecting the optimal expansion point is a non-trivial problem. We can use the closed captions to find sentence boundaries, as well as visual features to detect shot boundaries.

3. Conclusions and Future Work

Domain knowledge in the form of a small set of key phrases can provide semantic content about a TV series, and when used in conjunction with a data-driven approach to align the structure across episodes, this can help with chaptering each episode. This was demonstrated successfully for the TV show “Chopped”, where we built a query engine to search for food phrases across episodes. We would like to generalize the approach to other structured shows such as “Shark Tank”, “American Ninja Warrior” among others. Also, using video information such as shot-boundary and audio information such as speaker diarization can help with finer segmentation of relevant scenes (Knyazeva et al., 2015).

References

Aoki, Hisashi. High-speed topic organizer of tv shows using video dialog detection. *Systems and Computers in Japan*, 37(6), 2006.

Knyazeva, Elena, Wisniewski, Guillaume, Bredin, Hervé, and Yvon, François. Structured prediction for speaker identification in tv series. In *Interspeech 2015, 16th Annual Conference of the International Speech Communication Association*, Dresden, Germany, September 2015.

Manning, Christopher D., Surdeanu, Mihai, Bauer, John, Finkel, Jenny, Bethard, Steven J., and McClosky, David. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60, 2014. URL http://www.aclweb.org/anthology/P/P14/P14-5010.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL http://arxiv.org/abs/1301.3781.

Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Sys., Man., Cyber.*, 1979.

Sargent, Gabriel, Hanna, Pierre, and Nicolas, Henri. Segmentation of music video streams in music pieces through audio-visual analysis. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 724 – 728, 2014.

Smith, R. An Overview of the Tesseract OCR Engine. In *Proc. Ninth Int. Conference on Document Analysis and Recognition (ICDAR)*, pp. 629–633, 2007.

Yamamoto, Koji, Takayama, Shunsuke, and Aoki, Hisashi. Semantic segmentation of tv programs using corner-subtitles. In *2009 IEEE 13th International Symposium on Consumer Electronics*, pp. 205 – 208, 2014.