– Senior Thesis Final Report 2017/2018 –

Automatic Punctuation of Lecture Transcripts & Student Usage Analysis of Video Lectures in Online Learning Platforms

Ragy Morkos

Adviser: Ananda Gunawardena

Princeton University | Department of Computer Science

This thesis represents my own work and is in accordance with Princeton University's Honor Code and regulations.

Ragy Morkos

Abstract

Despite the prevalence of speech-to-text or automatic speech recognition (ASR) services, the discrepancy in the quality of the output transcripts can be large. Most notable in the vast majority of provided transcripts are their lack of punctuation or text segmentation. This can be commonly seen in automatically produced transcripts hosted on most online learning platforms, including YouTube. Despite Google (of which YouTube is a subsidiary) having shown to possess an engine for providing properly punctuated text in some of its other services, publicly available YouTube video transcripts nonetheless are still largely uncurated and are merely a stream of unpunctuated text. This can be especially problematic for YouTube video lectures and students who are using them. Students can depend on lecture transcripts for a variety of reasons, including hearing disabilities, students whose first language is not English, preference for easier note-taking, using them as checkpoints in videos or for search features, etc. While some of the online course providers who host their lecture videos on YouTube provide them with manually punctuated and curated transcripts, it is an expensive and tedious process and there still remains a large body of lecture content that does not have punctuated lecture transcripts.

This paper tries to explore an automatic solution for the stated problem of uncurated ASR transcripts. Through the training of a bidirectional recurrent neural network on manually punctuated lecture transcripts, we will show that it is possible to add punctuation to the transcripts with an overall F-score accuracy of 67%. We will be specifically implementing these new transcripts on Princeton Salon in COS 126, Princeton University's introductory CS class. Rolling out these newly punctuated transcripts to our student subject body, we did not notice an increase in student satisfaction despite the large improvement in the transcript quality. We therefore decided to conduct quantitative surveys as well as qualitative user studies/interviews to

better understand how students use transcripts and Salon in general in their studying. We conclude that context and organization of transcripts as well as quizzes and other features on an online learning platform matter much more than the quality of the transcripts based on our student responses. We furthermore propose a way to utilize student usage data on Salon to provide more organization and segmentation to transcripts as well as offer a preliminary analysis of student usage and interaction data on the Princeton Salon platform.

Contents

1. Introduction

The rise of online learning has certainly changed the landscape of education for good. A 2007 survey of more than 2,500 U.S. higher education institutions found that "online enrollments have continued to grow at rates far in excess of the total higher education student population" [1]. According to the same survey, institution boards "cite improved student access as their top reason for offering online courses and programs" with 83% of the institutions expecting their online enrollments to increase each year. The increase of online learning can be attributed to the overall positive results found when using it either as a substitute for the traditional classroom or as a secondary, helpful medium for learning as in the examples of online lecture videos or exercises. Indeed, a meta-analysis done by the US Department of Education of more than a thousand empirical studies of online learning concluded that "students in online learning conditions performed better than those receiving face-to-face instruction" [2].

Video lectures are perhaps the most important aspect of online learning [3, 4]. YouTube, especially, is a rich platform for the hosting of video lectures both by independent educators and larger online learning platforms such as Udacity[1] and Khan Academy[2] among others. One of the biggest benefits of videos hosted on YouTube is that they can offer automatically produced close-captioning functionalities. These close captions can be enormously helpful for a multitude of learners including, but not limited to, students with hearing disabilities and nonnative English speakers. Furthermore, on Princeton Salon, the platform hosting Princeton University's Computer Science online materials, transcripts are used as a way to search the video and as search points for navigating the video's different sections. Figure 1 shows an example of the

1 https://www.youtube.com/user/Udacity

2 https://www.youtube.com/user/khanacademy

transcripts used in Princeton Salon's videos taken from COS 226, Princeton's algorithms and data structures class.

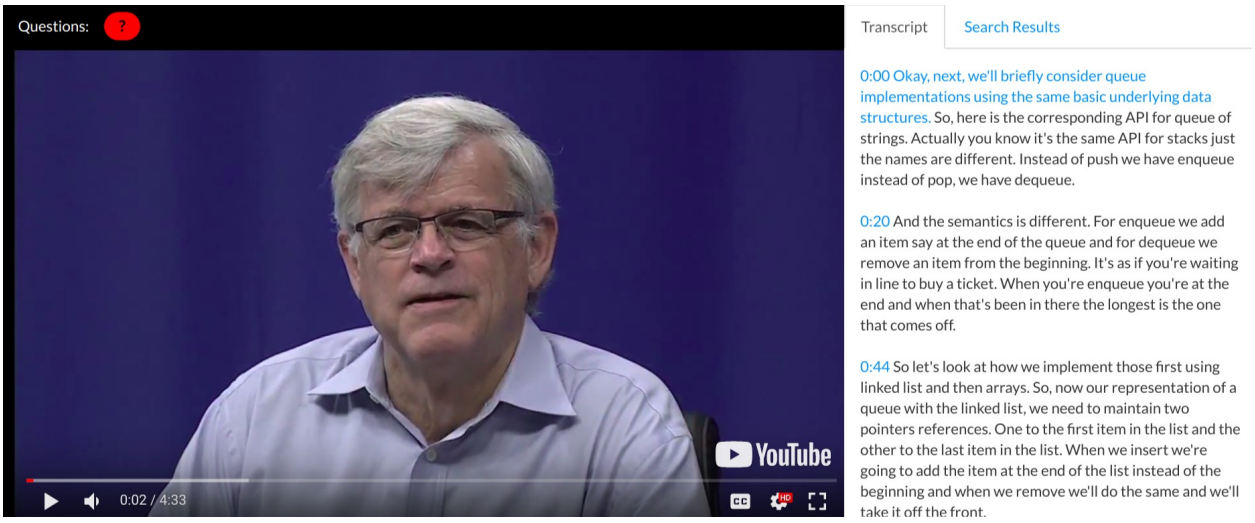

Figure 1: Screenshot from Princeton Salon showing the transcript integration

Despite their numerous use cases, from use as subtitles to integration into the lecture videos, the current automatically-generated close captions by YouTube lack any punctuation (the transcript shown in figure 1 was actually manually punctuated by Coursera since COS 226 is offered on their platform). While this might not be extremely problematic in their use as close captions or subtitles, it greatly delimits their use as transcripts to be deployed in a learning platform such as Princeton Salon. Moreover, because Princeton Salon aims to use transcripts for developing other learner-friendly features such as clustering, smart search, phrase mapping, word clouds, concept maps among many others, having properly punctuated transcripts is quintessential. This was our initial motivation for our endeavor to provide a way to automatically punctuate technical lecture transcripts / close captions produced from unpunctuated speech-to-text software output.

Our work on automatically punctuating YouTube's speech-to-text close captions is only one part of this paper. After the deployment of the newly punctuated transcripts on Salon for

COS 126, Princeton's introductory computer science class (which its transcripts were previously unpunctuated), we surveyed students to learn more about their usages of transcripts and to assess their satisfaction of the newly deployed transcripts. The results were surprising and informative and inspired us to conduct in-depth personal student user studies to better learn about how students use transcripts on Salon and how to make them better but also what they need in an online learning platform such as Salon. We propose a number of ways to address student needs on Salon and online learning platforms in general.

The remainder of this paper is organized as follows. In Section 2, we discuss related work and background, most notably previous work done on lecture video integration in online platforms and the state of automatic punctuation for automatically produced speech-to-text output. In Section 3, we discuss our approach to automatic punctuation, namely the technologies we considered and our leading presumptions about student usage. In Section 4, we discuss our implementation details, including how we gained data used for the machine learning part of our automatic punctuation endeavor, training techniques, failed attempts and possible reasons, as well as final methods used to achieve the results we will discuss at the end. We also discuss our techniques for learning about student usage patterns and behaviors both directly through anonymous and in-person surveys as well as data gathering through student interaction behavior gathered on the Salon platform. In Section 5, we will discuss the evaluation of the quality of our automatic punctuation mechanism. In Section 6, we will discuss proposed solutions and possible future work, mainly in furthering the quality of the automatic subtitles that is tailored towards lecture video content as well as other possible work that could be made towards improving automatic logical segmentation of different subtopics in a single lecture video. Finally, we conclude in Section 7, with a detailed Appendix in Section 9 for our implementation.

2. Background and Related Work

2.1 Lecture Videos and Transcripts in Online Learning Platforms

There have been numerous studies conducted on the usefulness of video lectures for students either as a substitute or a complement to in-class instruction [1, 2, 3, 4]. However, we will only be discussing features of video lectures, specifically transcripts and their integration with the video lecture content. In the specific case of transcripts, one study found that "the navigability of the lectures given by some kind of semantic markups was useful in allowing students to quickly find the relevant sections, in case they only wanted to review a particular portion of some lecture" [3]. Another paper focused on an efficient way of either manually or automatically producing "video digests," which basically entails having a lecture video divided into smaller video modules, each with its own summary and title [5]. In that paper, lecture transcripts were the focal key in producing these video digests, yet the paper presumed that transcripts are to be manually produced (at least editing-wise from ASR softwares) first before feeding into the video digest production software. Through our automatic punctuation mechanism specifically tailored to lecture content, we will offer a way to make the process even easier for production of video digests and for the production of automatically punctuated transcripts in general. Overall, having punctuated transcripts is important for any machine learning method deployed on the transcripts, as mentioned earlier and shown in [8, 9].

2.2 Related Work on Automatic Punctuation

There has been a fair amount of work done on both automatic punctuation for unpunctuated text (off-line processing) as well as punctuated speech to text output (on-line processing). In all the research literature we have seen, off-line punctuation processing seems to yield significantly higher accuracy than when the online speech-to-text recognition and

punctuation stages are combined [6, 7, 8, 9]. This makes intuitive sense as correct punctuation requires knowledge of the whole sentence or at least future nearby words in the sentence to be accurate enough. Reviewing multiple approaches to off-line automatic punctuation mechanisms, we have seen that neural networks for the task of punctuation seem to perform the best out of their competitors [10, 11, 12, 13, 14, 15, 16]. We will be specifically using a recurrent bidirectional neural network for the task of providing our automated punctuation [17].

2.3 Related Work on Obtaining Student Feedback and Usage Patterns

A number of papers lauding the importance of obtaining and analyzing student feedback have been made [23, 24, 25, 26, 27, 28]. Obtaining student feedback regarding transcripts and the Salon platform in general is quintessential to improving the platform and assessing the usefulness of new or improved features (in our case the punctuated transcripts). The two major ways for *directly* gaining information about student usage and feedback are surveys and in-person questionnaires. Surveys, usually anonymous, while ensuring truthful answers, are limited and simplistic for going in-depth into user feedback and their experiences using a product. In-person questionnaires solve that problem but are also liable to have students not be entirely truthful either consciously or unconsciously. Because both surveys and in-person questionnaires have their benefits, we have decided to conduct both as described in sections 3.2 and 4.2.

Another way for *indirectly* assessing and analyzing student usage is through data analytics or data gathering on the online platform (in our study, the platform is Salon). Numerous research has shown the benefits of analyzing student usage and interaction patterns on the platform and plotting out patterns and making sense of the data [29, 30, 31, 32, 33, 34]. We will show in section 4.3 our analysis of student interaction data on the Salon platform.

3. Approach and Design

3.1 Approach to Obtaining Curated Transcripts

Obtaining a large corpus of *relevant* training data was the most important aspect for training our neural network to be used in the punctuation task. Because we were trying to tailor our neural network's performance to COS 126 transcripts, our first approach was to train it solely on COS 226's manually edited transcripts. COS 226 is Princeton's algorithms class that is also available on Coursera, and so its transcripts have been manually edited and punctuated. Moreover, because both classes share the same instructor, the speech patterns between the transcripts of both classes would be identical. The only negative downside to the sole use of COS 226 transcripts was that we had a very limited amount of them, less than 2MB's worth. Because of this, training the neural network on such a small amount of data resulted in atrocious accuracy levels. Even when using prosodic features[3] (silence periods between words are the most notable), the neural network's performance was still atrocious, with an F-score less than 2%. Because of this, we realized we could not depend on such a small corpus of transcripts alone, even if it was incredibly similar to our end goal's transcripts we were trying to punctuate.

Our second approach was to use a pre-trained recurrent neural network trained in [17] to see the performance of a generically trained neural network. This neural network was trained on the European Parliment's transcripts[4], which have a size of about 324MB. While the size of the training corpus is very large, the accuracy of the punctuation was fairly poor, giving on average a 35% F-score for all of the punctuation marks when tested on COS 226's transcripts. We concluded that the relevancy between lecture transcripts, especially for technical content, and

3 This was implemented using Pydub: https://github.com/jiaaro/pydub

4 http://hltshare.fbk.eu/IWSLT2012/training-monolingual-europarl.tgz

generic speech such as the one found in the European Parliment's transcripts, was too little to produce a good neural network for punctuation restoration.

Our third approach was to find the next best training corpus we could find. The key idea is to keep the training transcripts as relevant as possible to COS 126's transcripts yet still be of large enough size and to have been already manually punctuated. Moreover, ease of obtaining the transcripts was also a factor since we were trying to obtain a large corpus of transcript data and so efficiency was important. We finally decided on using Udacity's manually produced and curated transcripts for our training corpus. The style of Udacity lectures closely resembles that of COS 126 and COS 226 including monologue lectures, technical content, among others. Using YouTube-dl[5], an open-source scraping tool tailored to YouTube and other media websites, we scraped and downloaded all of Udacity's English and manually edited and punctuated transcripts. After parsing and removing time-stamps, the Udacity transcripts came to about 40MB worth of punctuated text. In combination with COS 226's punctuated transcripts, our total training corpus came to about 41MB, with 1MB of COS 226's transcripts set aside for testing. As we shall see in Section 5.1, this training corpus's size was enough for a relatively high level of accuracy of automatic punctuation. We will discuss our implementation details of the recurrent bidirectional neural network in Section 4.1.

3.2 Approach to Getting Student Feedback

Using a simple online survey rolled out to COS 126 students, our aim was to increase the student response while still obtaining as much information as possible. We therefore opted to have only 3 simple questions in our survey to increase participation. For more in-depth review of how students use Salon and the transcripts, we conducted user studies with ten randomly selected

5 https://github.com/rg3/youtube-dl

students to obtain more information. Students were asked in a casual setting about their usage of Salon and their studying patterns in general and were awarded with a $10 Amazon gift card.

4. Implementation Details

4.1 Implementation Details for Neural Network Training

Our implementation of our recurrent bidirectional neural network (with an attention mechanism for punctuation restoration) is based on the work of [17], which is open-source and publicly available[6]. Using [17]'s model, we trained the neural network on our obtained training corpus from Section 3.1 on Princeton University's high-efficiency computation clusters. The input, however, has to be parsed such that punctuation marks are separated as tokens from the words. After parsing of the training corpus, it was ready to be used in our training stage. Figure 2 shows the flow of this process.

The training stage took 3 days using Theano[7] and Tmux's open source project[8] for background terminal usage on the computation clusters. The output is a binary file that is to be used with a Python script and the unpunctuated input text (in our case COS 126's transcript) to produce punctuated output. Output is tokenized (each token is a punctuation mark) and so it was ran through another Python script to remove the tokens and replace them with the punctuation marks they resemble. A Python testing script for calculating the F-score of the new output was also used to quantitatively assess the accuracy of our neural network. The Appendix in Section 9 deals extensively with the details for the implementation. The new COS 126 transcripts were rolled out on Princeton Salon after the first half of the Spring 2018 semester.

---

6 https://github.com/ottokart/punctuator2

7 https://github.com/Theano/Theano

8 https://github.com/tmux/tmux

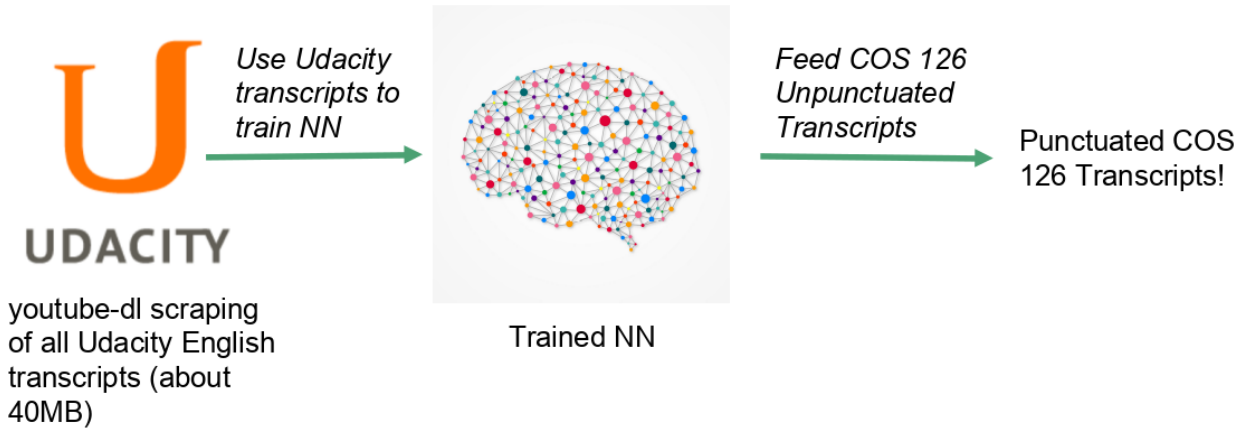

Figure 2: Representation of Process to Train Bidirectional Neural Network

4.2 Implementation Details for Obtaining Student Feedback

To obtain student feedback, we rolled out a very simple 3 question survey to the students on Piazza. A screenshot from the survey can be found in Figure 3.

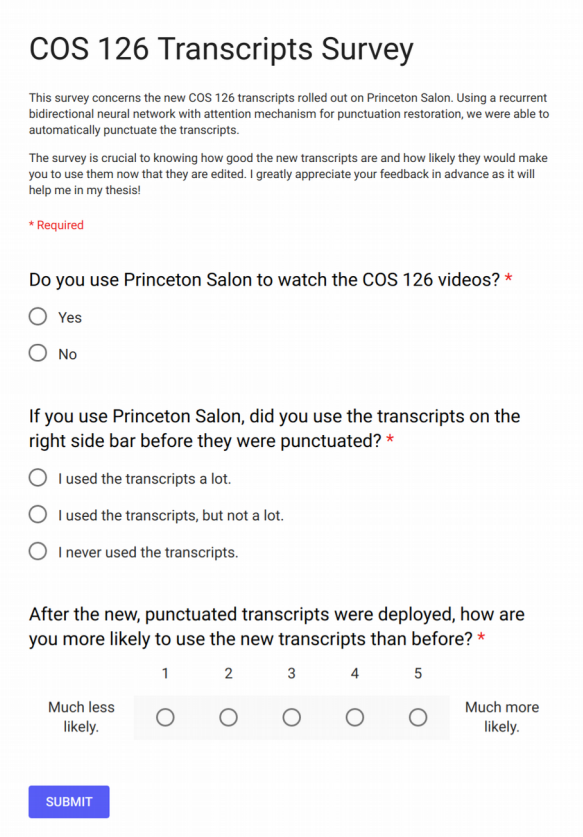

Figure 3: Survey Sent to COS 126 Students for Feedback about New Transcripts

The purpose of the survey was to gauge how useful the new transcripts were for the students by asking them first whether or not they use the Princeton Salon website and how frequently they would use the website both before and after the transcripts were punctuated.

The in-depth student user studies went much more in-depth about how the students used the old and new transcripts as well as their usage of Salon and online learning platforms in general. Below are some of the main questions asked to students participating in the in-person user studies:

- *Do you describe yourself as a "techy" person or love the integration of technology into everyday stuff?*
- *Before COS 126, were you an avid user of online platforms like Coursera, edX, Khan Academy, etc?*
- *Do you watch all the video lectures for COS 126 or do you sometimes skip them?*
- *Do you watch them on Salon or on YouTube directly?*
- *Do you continue watching the video till the end?*
- *Do you skip parts of the video a lot?*
- *Do you watch at a higher/lower speed?*
- *Do you use the transcripts feature on Salon?*
- *If you use the transcripts, did you notice they were punctuated during the second half of the semester?*
- *If you did notice they were punctuated, did that make a difference?*
- *Do you find anything especially frustrating regarding the transcripts or their integration on Salon?*

- *If the transcripts were segmented/separated into different parts (Intro/Background, Concept, Implementation, Analysis, etc.), would you be more likely to utilize them?*
- *How do you think the transcripts could be integrated better on Salon?*
- *Do you have other suggestions for improvement for the transcripts or Salon in general?*

4.3 Implementation Details for Analyzing Student Usage on Salon

Evaluating student usage through analysis of their usage of the Salon platform is also a very important aspect of this paper. On Salon, student usage was tracked in increments of 5 seconds, with relevant information captured for each data point as shown in Figure 4. The gathered data evaluated to close to 300 megabytes worth of data points across students enrolled in COS 126 and COS 226 at Princeton. Parsing the user data was implemented using different Python scripts for obtaining specific segments or intersections of the user interaction data. The gathering and analysis of this data is important as it shows how students actually use the Salon platform and the video lectures.

```
Date | Time | Start Time (seconds) | Playback | YouTube Video ID | User
```

Figure 4: Structure of Student Usage Data Gathered on Princeton Salon

5. Evaluation Results

5.1 Performance of Trained Neural Network on Lecture Transcripts

As mentioned in Section 3.1, we had set aside 1MB worth of punctuated transcripts from COS 226 to be used in testing the accuracy of our neural network model. The transcripts are

copied and stripped from their punctuations and passed to our neural network to be punctuated.

The results of our neural network punctuation are shown in Table 1 below:

Punctuation Mark	Precision	Recall	F-score
Comma	57.7	45.9	51.1
Period	72.3	72.5	72.4
Question Mark	65.5	57.9	61.5
Semicolon	98.3	73.5	84.1
Overall	*73.5*	*62.5*	*67.3*

Table 1: Statistics of Neural Network Performance

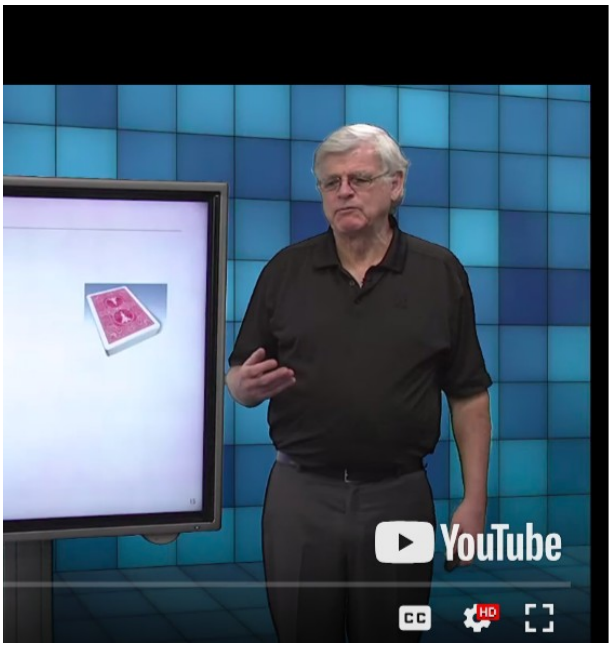

Figure 5: Screenshot of Salon After Deployment of Punctuated Transcripts

As one can see from Table 1, the average F-score for punctuation detection was 67.3%. The F-score for each punctuation mark varied greatly; semicolons and periods were restored with

a very high F-score, 84.1% and 72.4%, respectively. This is in contrast to the punctuation restoration of commas and question marks, which their F-scores came to 51.1% and 61.5%, respectively. These results make intuitive sense as periods and semicolons generally depend less on context and more on sentence boundary detection as opposed to commas and question marks. A screenshot of the new transcripts shown in Figure 5 (automatically punctuated COS 126 transcripts) can be compared with that of figure 1 (manually punctuated COS 226 transcripts); as seen, the quality of the automatically punctuated transcripts rose considerably.

5.2 Student Usage Analysis on Salon

Student Usage of the videos on Salon was analyzed in order to obtain useful information about student interaction on the platform. Figure 6 shows the student watches at each 5 second increment across all COS 126 videos. We can clearly see from the graph that there is an exponential decrease in students watching the videos as time progresses. The most likely reason for this significant decrease is that students do not watch a video till the end. This can be due to loss of interest, prior familiarity with the concept(s), among others.

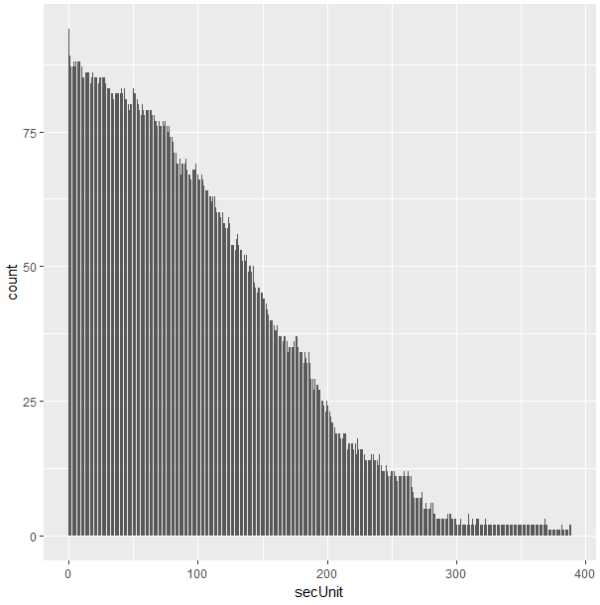

Figure 6: Student Watching Counts in 5 Second Increments

Further analysis into student usage by date also yielded some interesting results. As seen in Figure 7, student viewing of COS 126 lecture videos across the semester was in seemingly random spurts rather than uniform across different dates, even sequential ones. We can attribute the really large spurts of video views to midterms and finals, but further conclusions are harder to draw. We will come again to these results in Section 6.

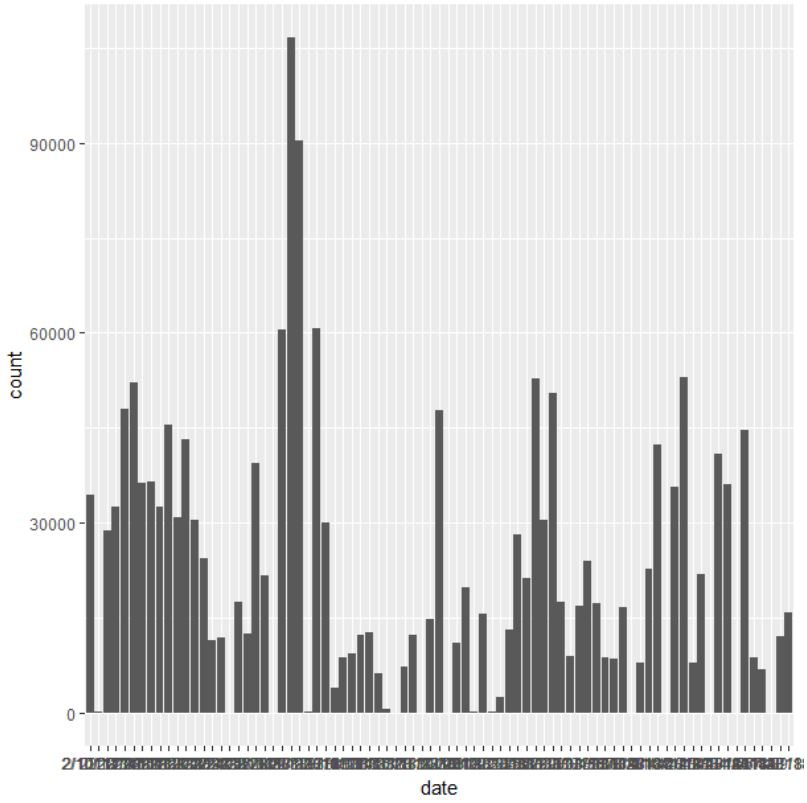

Figure 7: Student Watching Counts Across Sequential Dates

5.3 Student Feedback on Curated Transcripts and Salon

Student feedback on the transcripts shed light on a number of important issues. Out of the 31 COS 126 students who answered our survey, 90.3% of students used Salon. 71% of the students in the survey did not use the transcripts before they were punctuated. After punctuation, 67.7% of students were neutral to the newly punctuated transcripts while 29% were more likely

to use them than before they were punctuated, signifying a small increase in usability due to the punctuation. Despite the promising results in the student survey, they were not quite conclusive in understanding how the transcripts would be useful in general and how they fit within the usage patterns of students in general. To that end, we conducted in-depth interviews with ten randomly selected students as explained in Section 4.2, and the results shed light on how students use the Salon platform and their study patterns in general.

Most students interviewed did not even notice the roll-out of punctuated transcripts in the second half of the semester, even the students who claimed to be users of the transcripts feature on Salon. This was surprising in that although the newly punctuated transcripts were of a much higher quality, students did not necessarily notice them. Moreover, some students said that because the lecturer was speaking at a very slow pace, they would increase the video speed to 1.25x or 1.5x, rendering the use of transcripts an unnecessary distraction due to how fast they would roll on the screen. Students expressed significant interest in having the transcripts segmented into different topics (background, main concept(s), implementation, etc.) as well as have quizzes and much more questions integrated within the transcripts section.

6. Future Work

6.1 Future Work in Automatic Punctuation

There are a number of possible improvements regarding automatic punctuation. Our research shows the incredible importance of relevancy and context-similarity to training a neural network for punctuation restoration; despite our smaller training corpus of mostly Udacity transcripts (40MB) compared to the European Parliment's transcript corpus (324MB), the neural network trained on the former performed almost twice as better than the one trained on the latter.

Future work for improving the accuracy of our automatic punctuation model involves obtaining a larger training corpus of lecture transcripts. Because Udacity's transcripts are available on YouTube, scraping them off was relatively easy and time-efficient given their amount. However, a lot of lectures and their associated transcripts are hosted outside of YouTube and so they would require writing specialized scripts for each MOOC provider. These MOOC providers include, but are not limited to, edX, Coursera, MIT Open Courseware, among others.

Moreover, the use of prosodic features was not implemented in this paper. However, prior work has shown that the combination of lexical and prosodic features when training a neural network can result in higher accuracies than when training using only one of each [21, 22]. This, however, requires a dedicated setup as the audio files needed for the use of prosodic features in punctuation restoration require terabytes of storage capacity.

6.2 Future Work in Topic Segmentation

One of the notable ideas for transcript features that interviewed students agreed were important was having the transcripts segmented into different subtopics. For example, in the case of a COS 126 video, subtopics could be in the form of "Background/Introduction", "Concept", "Implementation" (code in Java), and "Analysis" (in the case of algorithms, this could be runtime). This is a very generic example; each COS 126 video will be different and will require different subtopic titles. Interviewed students, most of whom mentioned that they find no use for the transcripts, mentioned that such segmentation would be helpful.

While possibly helpful, logical segmentation of text into paragraphs with different subtopics is a harder task than that of automatic punctuation since for punctuation after each word, there is only a handful of options. For logical segmentation, however, the decision to transition after a sentence into a different paragraph or subtopic is a much harder problem and

requires different properties to consider. Numerous approaches to this problem have been made including "TextTiling" [18], decision tree and hidden Markov modeling techniques [19], combination of sentence features (key words, punctuation, quotations, etc.) to be used in machine learning classification tasks [20], among others. Each approach has its own pros and cons, the biggest of which are having a supervised vs. unsupervised techniques as one requires a fairly large training set and usually yields a higher accuracy while the other doesn't require a dataset but can be less accurate.

Logical segmentation of text can be approached in a number of ways. Although our endeavors to offer a valid solution to the problem was not fruitful, we have still opted to include our approaches here for educational purposes. Our original intent was to offer a new and functional approach for logical segmentation inspired by the Salon user interaction data. It is important to note that for most logical segmentation processing to be useful, adequate punctuation should be present, and hence, our work on automatic punctuation for lecture transcripts discussed before would prove to be useful in this regard.

Our first trial in doing this in COS 126/226's punctuated transcripts was to detect paragraph segmentation via shift in video frames. More specifically, for COS 126/226 video lectures, this would manifest in video frame change between presentation slides and the instructor. Our initial hypothesis was that this shift in frame within the video is an indicator of a shift in context, therefore warranting a different paragraph. Unfortunately, this yielded highly inconsistent results for all the thresholds of change we have tried. We concluded that this approach of detecting video frame change as an indicator of context change is a poor one.

Our second approach aimed at using aggregated student usage data from Princeton Salon in order to detect different subtopics in each lecture video module. The structure of the aggregated student usage data from Princeton Salon was shown previously in Figure 4. Data

points in the dataset were taken every five seconds. Using this data in aggregate, we can see for each video on average which segments students would specifically choose to watch. Usually, students refer to certain portions of the videos depending on what they want to review. This resembled a perfect opportunity to crowdsource this usage data and be able to infer as much as possible subtopics within each lecture video. Unfortunately, the segmentation obtained through using this method did not yield accurate results.

7. Conclusion

With the continuous high growth in online learning and the use of lecture videos and the flipped classroom model, improving the integration and quality of video lectures is of great importance. Punctuation of lecture transcripts is not only helpful for users of the transcripts, but they also open up ways for the processing of the transcripts for logical segmentation, meaning extraction, among other machine learning endeavors. Our contribution in this paper has been the establishment of a high-accuracy neural network model that would be catered to the punctuation of ASR output of lecture videos. Moreover, our analysis of student usage both through user interaction data gathered by the Salon platform as well as surveys and in-person user studies have shed light on how students actually interact with lecture videos and what features, including transcripts, matter most to them and would enhance their learning.

8. References

[1] Allen, I. Elaine, and Jeff Seaman. Online nation: Five years of growth in online learning. Sloan Consortium. PO Box 1238, Newburyport, MA 01950, 2007.
[2] Means, Barbara, et al. "Evaluation of evidence-based practices in online learning: A meta-analysis and review of online learning studies." (2009).
[3] Ronchetti, Marco. "Using video lectures to make teaching more interactive." International Journal of Emerging Technologies in Learning (iJET) 5.2 (2010): 45-48.
[4] Brecht, H. David. "Learning from online video lectures." Journal of Information Technology Education 11.1 (2012): 227-250.
[5] Pavel, Amy, et al. "Video digests: a browsable, skimmable format for informational lecture videos." UIST. 2014.
[6] Chen, C. Julian. "Speech recognition with automatic punctuation." Sixth European Conference on Speech Communication and Technology. 1999.
[7] Kim, Ji-Hwan, and Philip C. Woodland. "The use of prosody in a combined system for punctuation generation and speech recognition." Seventh European conference on speech communication and technology. 2001.
[8] Makhoul, John, et al. "The effects of speech recognition and punctuation on information extraction performance." Ninth European Conference on Speech Communication and Technology. 2005.
[9] Matusov, Evgeny, Arne Mauser, and Hermann Ney. "Automatic sentence segmentation and punctuation prediction for spoken language translation." International Workshop on Spoken Language Translation (IWSLT) 2006. 2006.
[10] Huang, Jing, and Geoffrey Zweig. "Maximum entropy model for punctuation annotation from speech." Seventh International Conference on Spoken Language Processing. 2002.
[11] Batista, Fernando, et al. "Bilingual experiments on automatic recovery of capitalization and punctuation of automatic speech transcripts." IEEE Transactions on Audio, Speech, and Language Processing 20.2 (2012): 474-485.
[12] Liu, Yang, et al. "Enriching speech recognition with automatic detection of sentence boundaries and disfluencies." IEEE Transactions on audio, speech, and language processing 14.5 (2006): 1526-1540.
[13] Beeferman, Doug, Adam Berger, and John Lafferty. "Cyberpunc: A lightweight punctuation annotation system for speech." Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on. Vol. 2. IEEE, 1998.
[14] Shen, Wenzhu, et al. "Automatic punctuation generation for speech." Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on. IEEE, 2009.
[15] Shriberg, Elizabeth, and Andreas Stolcke. "Direct modeling of prosody: An overview of applications in automatic speech processing." Speech Prosody 2004, International Conference. 2004.
[16] Batista, Fernando, et al. "Recovering punctuation marks for automatic speech recognition." Eighth Annual Conference of the International Speech Communication Association. 2007.
[17] Tilk, Ottokar, and Tanel Alumäe. "Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration." Interspeech. 2016.
[18] Hearst, Marti A. "TextTiling: Segmenting text into multi-paragraph subtopic passages." Computational linguistics 23.1 (1997): 33-64.
[19] Salton, Gerard, et al. "Automatic text structuring and summarization." Information Processing & Management 33.2 (1997): 193-207.

[20] Sporleder, Caroline, and Mirella Lapata. "Broad coverage paragraph segmentation across languages and domains." ACM Transactions on Speech and Language Processing (TSLP) 3.2 (2006): 1-35.
[21] Kolář, Jáchym, Elizabeth Shriberg, and Yang Liu. "Using prosody for automatic sentence segmentation of multi-party meetings." International Conference on Text, Speech and Dialogue. Springer, Berlin, Heidelberg, 2006.
[22] Kolář, Jáchym, and Lori Lamel. "Development and evaluation of automatic punctuation for French and English speech-to-text." Thirteenth Annual Conference of the International Speech Communication Association. 2012.
[23] Richardson, John TE. "Instruments for obtaining student feedback: A review of the literature." Assessment & evaluation in higher education 30.4 (2005): 387-415.
[24] Leckey, Janet, and Neville Neill. "Quantifying quality: the importance of student feedback." Quality in Higher Education 7.1 (2001): 19-32.
[25] Centra, John A. "Effectiveness of student feedback in modifying college instruction." Journal of Educational Psychology 65.3 (1973): 395.
[26] Hounsell, Dai. "Student feedback, learning and development." Higher education and the lifecourse (2003): 67-78.
[27] Watson, Sarah. "Closing the feedback loop: Ensuring effective action from student feedback." Tertiary Education & Management 9.2 (2003): 145-157.
[28] Nathenson, Michael B., and Euan S. Henderson. Using student feedback to improve learning materials. Routledge, 2018.
[29] McMillan, James H., and Sally Schumacher. "Research in Education: Evidence-Based Inquiry, MyEducationLab Series." Pearson (2010).
[30] Zhuoxuan, Jiang, Zhang Yan, and Li Xiaoming. "Learning behavior analysis and prediction based on MOOC data." Journal of computer research and development 52.3 (2015): 614-28.
[31] Guo, Philip J., Juho Kim, and Rob Rubin. "How video production affects student engagement: An empirical study of mooc videos." Proceedings of the first ACM conference on Learning@ scale conference. ACM, 2014.
[32] Daradoumis, Thanasis, et al. "A review on massive e-learning (MOOC) design, delivery and assessment." P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on. IEEE, 2013.
[33] Jiang, Suhang, et al. "Predicting MOOC performance with week 1 behavior." Educational Data Mining 2014. 2014.
[34] Zhuoxuan, Jiang, Zhang Yan, and Li Xiaoming. "Learning behavior analysis and prediction based on MOOC data." Journal of computer research and development 52.3 (2015): 614-28.

9. Appendix

Appendix A

Scraping ASR YouTube Transcripts from Videos/Channels

There are numerous ways to scrape transcripts for use in machine learning methods (in our example, the training of a recurrent bidirectional neural network) or data analytics. The easiest and most efficient way is to use `youtube-dl`, a command-line open-source framework for extracting media formats (text, audio, or video) from YouTube and other media sites. Below is the command we used to extract transcripts from Udacity's channel on YouTube (assuming youtube-dl is installed on the system).

```
youtube-dl --all-subs –skip-download https://www.youtube.com/user/Udacity
```

Running this command took approximately 12 hours for the size of Udacity's YouTube channel (29,000+ videos as of May 4th, 2018). The `--all-subs` flag downloads subtitles for all languages. Although `youtube-dl` does possess flags for specifying the language of transcripts to be downloaded, the subtitles for the same language can be under different categories. For example, English transcripts can be put under `eng` or `eng-hi` (HI standing for hearing-impaired). Because of this, we opted to scrape all of the transcripts for all the videos on Udacity's YouTube channel and take a look at the downloaded files to figure out the header names for all the English language ones.

The `-skip-download` flag skips the download of each video file and downloads its transcript only. Although we did not utilize lexical features for the neural network training, we would have kept this flag should we have opted to utilize lexical features. It is important to note, however, that simply removing the `-skip-download` flag would be problematic as this will download along with the transcripts the original video files, which would require an enormous

amount of storage. To this end, if we were to download the audio, we would have to not only remove the `-skip-download` flag, but also add the `-extract-audio` flag as well to download the audio of each video only, as follows:

```
youtube-dl --all-subs -extract-audio https://www.youtube.com/user/Udacity
```

Appendix B

Removing non-English Transcripts

As mentioned in Appendix A, downloading the transcripts in all languages then removing non-English ones afterwards is the best approach to make sure all English language transcripts are obtained. The next two steps are to remove any non-English transcripts and removing timestamps from the remaining transcripts. Appendix B deals with removing the non-English transcripts.

All transcripts scraped using `youtube-dl` by default possess a language header at the end of the file. For example, an English transcript would be named `VideoName-eng.srt` with the `eng` header describing that this is an English transcript. Similarly, a French transcript would be named `VideoName-fre.srt` with the `fre` header describing that this is a French transcript. Knowing this structure obtained through `youtube-dl` scraping, there are a number of ways to approach removing non-English transcripts, the easiest of which is to just utilize Linux's powerful command line tools. Using the Linux `rm` command, we can simply call the following command in a Linux shell (inside the directory where the transcripts are hosted) to delete transcripts with a certain language:

```
rm *fre.srt
```

We will need to do this for every language we do not need for the training of our neural network. It is important to note that although tedious, this approach is better than deleting all the transcripts without a certain language header in the filenames. Because YouTube stores its English transcripts with different headers as mentioned previously (i.e. American English,

British English, English with Hearing Impaired subtitles, etc.), we will need to eliminate the languages of the transcripts one by one until we are left with the English transcripts only.

Appendix C

Removing Time-stamps from Transcripts

YouTube transcripts come in the following format example:

```
01:23 This is an example text.
```

To utilize the transcripts as training data for machine learning methods (in our example, for a recurrent bidirectional neural network), we will first need to remove all the time-stamps from the transcripts. A simple Python script can be used to perform this task:

```
path = # path to transcripts
filenames = sorted(os.listdir(path))
for i in range(len(filenames)):
    s = ""
    with open(os.path.join(path, filenames[i]), 'r') as f_original,
open(os.path.join(path, filenames[i][:-5] + "_no_time-stamp.txt"), 'w') as f:
        for line in f_original:
            s += line.strip()[6:] + '\n'
        f.write(s)
```

This simple script assumes that the first 6 characters of each line in the transcript are reserved for the time-stamp. We simply remove those 6 characters on each line and write to a new file without the time-stamp. It is important to note that this format could be different depending on the method used to scrape the transcripts from YouTube, or, more broadly, other mediums where lectures are hosted, since MOOC providers such as Coursera, edX, among others, all have their own platforms. Therefore, this simple script should be tweaked depending on the format.

Appendix D

Training the Recurrent Bidirectional Neural Network

As mentioned in section 4.1, our recurrent bidirectional neural network infrastructure is adapted from [17]. The stages for the training can be divided into four main steps:

1) Replace punctuation in punctuated transcripts with tokens.

2) Use the parsed transcripts for training (computationally intensive task)

3) Use the trained neural network to punctuate needed transcripts (output contains tokens)

4) Remove tokens from output transcripts.

The input punctuated transcripts need to be in the following format for use in the training of the neural network:

```
to be ,COMMA or not to be ,COMMA that is the question .PERIOD
```

As shown in the example above, all punctuation marks should be stripped and replaced with a token (for example the token for a comma is `,COMMA`). Moreover, because casing of words could interfere with the training process (for example, the word "I" is capitalized, yet can be present in the middle of the sentence, leading to the "confusion" of the neural network in the training process). To this end, we have to convert all the training text into lower casing in addition to replacing punctuation marks with tokens. The following Python script can be used to perform this task and will output the training text into one file:

```
import os

path = # path to folder where transcripts live

# variable holding the transcripts stripped of time-codes
string = ''

# iterate over the list getting each transcript file
for s in sorted(os.listdir(path)):
    filee = path + '/' + s

    # open the file and then call .read() to get the text
    with open(filee) as f:
        # split by new line
        text = f.read().split('\n')
        for i in range(0, len(text), 5):
            if i + 3 < len(text): # in case there's no lines
                string += text[i + 2].strip('\r')
                string += ' '
            if i + 4 < len(text): # in case there's only one line
                string += text[i + 3].strip('\r')
                string += ' '
        string += '\n'

# remove capitalizations from the training transcripts so the model doesn't get hints
string = string.lower()

# At this point we have the transcripts split into paragraphs for each video.
# Now we are going to add the syllables of the punctuation. Example:
# to be ,COMMA or not to be ,COMMA that is the question .PERIOD

# function to handle adding syllables of punctuations
def add_punctuations(punctuation_mark, syllable, string):
    parsed_string = string.split(punctuation_mark)
    new_string = ''
    for i in range(len(parsed_string)):
        if i < (len(parsed_string) - 1):
            new_string += parsed_string[i]
            new_string += ' '
            new_string += syllable
            if punctuation_mark == '-':
                new_string += ' '
        else:
            new_string += parsed_string[i]
    return new_string

# add all the syllables to the transcripts
string = add_punctuations(',', ',COMMA', string)
string = add_punctuations('.', '.PERIOD', string)
string = add_punctuations('?', '?QUESTIONMARK', string)
string = add_punctuations('!', '!EXCLAMATIONMARK', string)
string = add_punctuations('-', '-DASH', string)
string = add_punctuations(':', ':COLON', string)
string = add_punctuations(';', ';SEMICOLON', string)

with open('training_data.txt', 'w' ) as f:
      f.write(string)
```

The second step of training the bidirectional recurrent neural network requires the access to powerful computational resources. The easiest option (and the most widely used) is to use a dedicated computation server for this task. We used the Cycle[9] computation servers of the Computer Science department at Princeton in order to train the neural network.

Most servers can be SSH'd into. However, because the training of the neural network takes a number of days, the SSH connection would have to be maintained for the length of the running time of the training task. Fortunately, Tmux[10] can help with this issue. A terminal multiplexer, Tmux can be used after connecting via SSH whereby a Tmux session is opened and the training process happens inside the Tmux session. Detaching from the Tmux session and even logging out of the server/cutting the SSH connection does not halt the training processing.

To train the neural network, we will need to have Theano, Numpy, as well as Python 2 installed on the server. Moreover, we will also need the neural network infrastructure from [17]. Within the main directory holding the files from [17], we will first call a Tmux session as follows:

```
tmux new -s session_name
tmux attach -t session_name
```

Once the `attach` command is initiated, we will be within a Tmux session. We can now train the neural network as following:

```
python main.py <model_name> <hidden_layer_size> <learning_rate>
```

9 https://csguide.cs.princeton.edu/resources/cycle

10 https://github.com/tmux/tmux

As mentioned earlier, this task can take between 3+ days, depending on the size of the training data. Since it is unlikely that the SSH connection can be kept for the period of the training task, we can detach from the Tmux session and logout from the SSH connection using the two following commands:

```
tmux detach
logout
```

The output from the training task will be a `.pcl` binary file that will host the neural network and will be used in subsequent stages.

Appendix E

Punctuating Transcripts and Time-stamp Restoration

After the training stage is finished and the `.pcl` file is outputted, we can then begin the punctuation stage for any unpunctuated transcripts we have. To do so, we will issue the following command:

```
cat unpunctuated.txt | python punctuator.py NN.pcl punctuated_with_tokens.txt
```

This will result in a punctuated file called `punctuated.txt`. However, because this file will still contain punctuation tokens instead of punctuation marks, we will have to pass this file through another Python script to replace the punctuation tokens with punctuation marks as following:

```
python convert_to_readable.py punctuated_with_tokens.txt punctuated.txt
```

The resulting file, `punctuated.txt`, will contain the punctuated transcripts sans tokens. However, there still remains the restoration of time-stamps back to the transcripts. The following simple Python script can be used to restore time-stamps back into the transcript:

```
import os
import string

with open('punctuated.txt', 'r') as f, open('original.srt', 'r') as f_original:
    punctuated = f.read().strip().split()
    stringg = ""
    counter = 0
    for line in f_original:
        line_split = line.strip().split()
        stringg += line_split[0]
        for j in range(len(line_split) - 1):
            if counter < len(punctuated):
                stringg += ' ' + punctuated[counter]
                counter += 1
            else:
                break
        stringg += '\n'
    with open('punctuated_timed.srt'), 'w') as new_f:
        new_f.write(stringg)
```

Note, that if more than one transcript needs to be punctuated, either a Bash script or simple Python script can be used to iteratively perform this process for each transcript.