# *The Office* Character Dialogue Generation Using Images

Team: Assistants (to the) Regional Programmer
Noah Foster, Chloe Griffin, Abubakarr Jaye, Christine Tseng
(nfoster3, dcgriffi, ajaye, ctseng6)

Brown University, Fall 2022 CS2470 Final Report

## 1 Introduction

In this project, we hope to build a dialogue generator for the television show *The Office* (2003). In particular, the dialogue generator will

1. take in an image, or still frame, from the show,
2. identify the present characters and objects within the scene, and
3. output lines based on what the characters would say in the given environment.

Thus, our project is like image captioning, but with a twist – we now grab context from the entire TV show's scripts to inform the "captioning" of a still frame through generating potential character dialogue.

There have been many attempts to create models that generate captions for images; however, there has not been any attempt to generate character-informed dialogue based on an image. This is interesting work since it can help screenwriters and manga writers generate content or ideas when creating storylines. The problem is thus a classification and structured prediction task. The model needs to classify the characters in the image and also generate dialogue text for each character.

## 2 Methodology

We will have two major network components in our model. The first network will conduct our image classification tasks, while the second network will be a language model that propagates our image classification results into character dialogue output.

### 2.1 Network I: Image Classification

For this network, we implement two image processing architectures: one to extract the character present in the scene, and another to extract the objects in the scene.

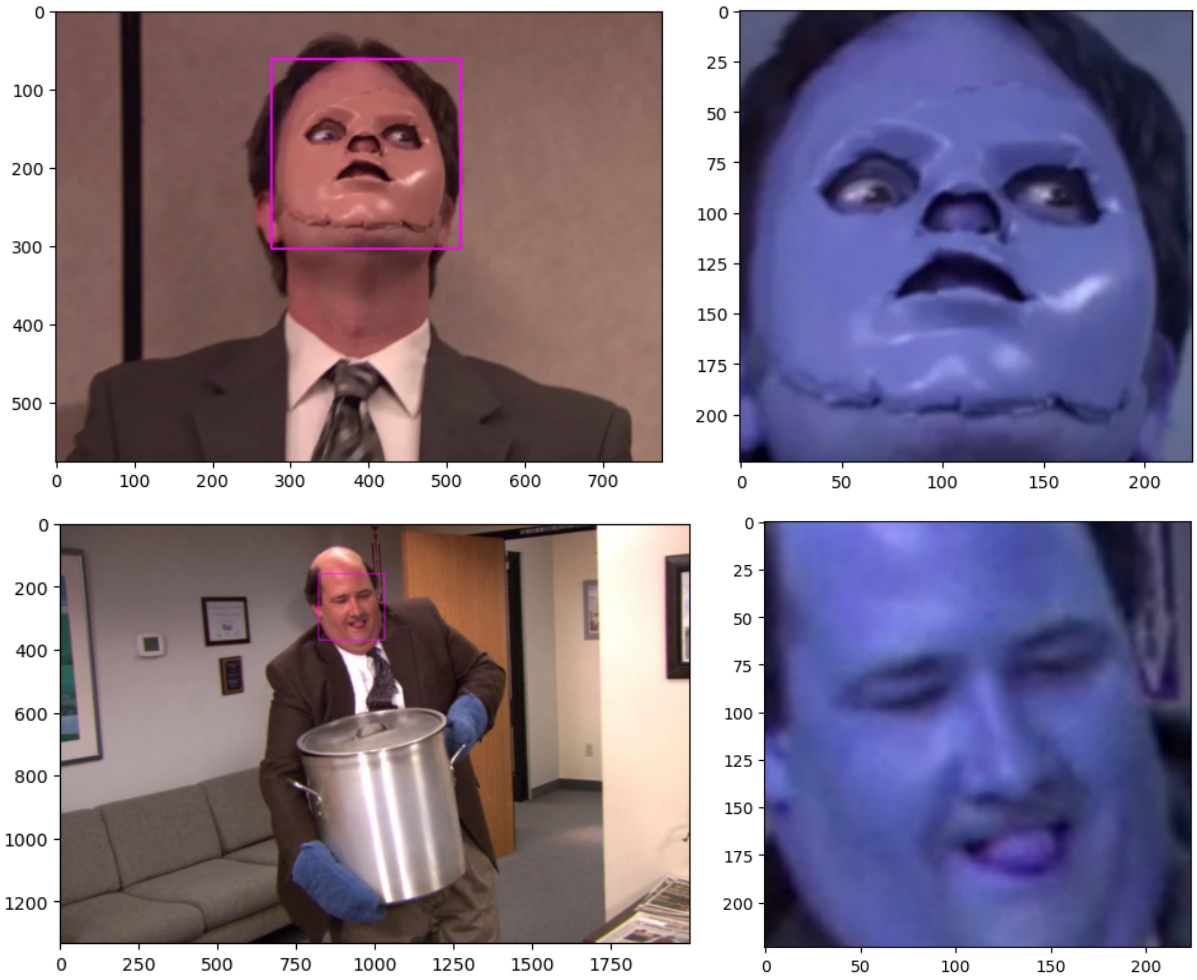


Figure 1: Examples of Preprocessed Images (Haar Cascade)

### 2.1.1 Character Recognition: VGG16 with vggFace2

In order to extract characters from still frames, we first employ preprocessing methods to clean our dataset for facial-recognition use. Our technique is made up of detecting a face within an image using the Haar cascade classifier [1], and then cropping and resizing the image to only the face present in the original image. Examples of the preprocessing are provided in Figure 1: we see how a face is detected and then cropped (and recolored, in order to intensify differences in facial features). These cropped images are thus fed into our character recognition model.

For our character recognition model, we build a custom model named The Office CNN. To do this, we use tranfer learning[1] on a VGG16 model pre-trained on the vggFace2 dataset [3], which is a dataset containing 3.1 million images of over 9000 subjects. The VGG16 model is a convolutional neural network with 16 layers. We thus re-train for classification on our *The Office* Characters [4] dataset, which contains images of six main characters from the show: Angela, Dwight, Jim, Kevin, Michael, and Pam. The output of which characters

[1]Our implementation pulls inspiration from this CODE Magazine article [2].

are present is the main informant of dialogue generation in the next network.

### 2.1.2 Object Detection: CLIP

To incorporate additional information about the scene of each still frame, we implement the zero-shot image classification method CLIP (Contrastive Language–Image Pre-training)[2] [7]. Using clip-vit-base-patch32 [5] , we are able to check (without training!) whether each still frame contains anything from a list of relevant nouns, provided in Table 1. Note that this list of nouns was generated by scraping through the entire *The Office* script, and then subsetting based on expert[3] knowledge.[4] With the object detection, we are able to consider the general environment of the image (i.e. outside, inside, time of day, etc.) or other objects in the image (office supplies, food, etc.) to inform our dialogue generation.

| **Relevant Nouns List** | | |
|---|---|---|
| agriculture | door | night |
| aircraft | drink | office |
| art | fire | paper |
| baby | fish | park |
| bar | food | party |
| bed | football | phone |
| bedroom | fruit | proposal |
| boat | gallery | railway |
| book | game | road |
| bottle | garden | sea |
| bread | group | snow |
| bridge | hospital | stapler |
| cabinet | hotel | street |
| car | house | table |
| cash | kitchen | telephone |
| cat | library | television |
| chair | lunch | theatre |
| church | machine | toilet |
| coffee | manager | train |
| company | milk | tree |
| computer | mirror | trophy |
| copier | money | tv |
| cup | mug | university |
| desk | museum | water |
| dog | newspaper | window |

Table 1: List of Relevant Nouns, scraped and subsetted from *The Office* script.

[2]Our implementation pulls inspiration from these Hugging Face [5] and OpenAI [6] tutorials.

[3]us — we are the experts on *The Office*.

[4]This list of objects/nouns could be easily changed by replacing the `nouns.csv` file in our GitHub repository (link the GITHUB!!!!.

With both character recognition and object detection, our image network is about to identify the present characters and objects within a still frame. Example of the output are provided in Figure 2.

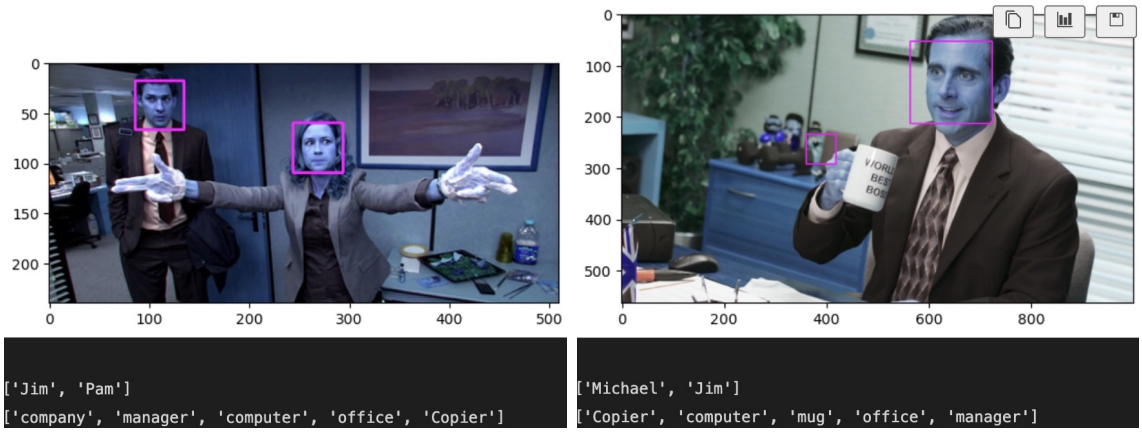



Figure 2: Examples of Image Network Output.

### 2.2 Network II: Language Model

For the second part of our model, we fine-tune[5] a GPT-2 model on the transcript of *The Office* [10]. We choose a pre-trained language model (LM) since models that we can train from start to finish on our own computers will not be able to output any dialogue that sounds remotely like coherent english, much less a specific character. In preparing the training data, we include all the characters in the scene, all nouns mentioned in the scene and the dialogue of the scene for each prompt[6].

## 3 Results

Given our pipeline is made of multiple networks, we provide in Tables 2 and 3 the training and accuracy metrics found within the various models of our architecture.

**Image Accuracy Metrics**

| Metric | The Office CNN | |
|---|---|---|
| Training | Accuracy | 0.9962 |
| Loss | Categorical Cross-Entropy | 0.0203 |
| Testing | Accuracy | 0.7428 |

Table 2: Accuracies from Character Recognition (VGG16)

[5]Our implementation pulls inspiration from these Hugging Face [8] and Kaggle [9] articles.
[6]Our implementation pulls inspiration from this Hugging Face [11] article.

**LM Training Metrics**

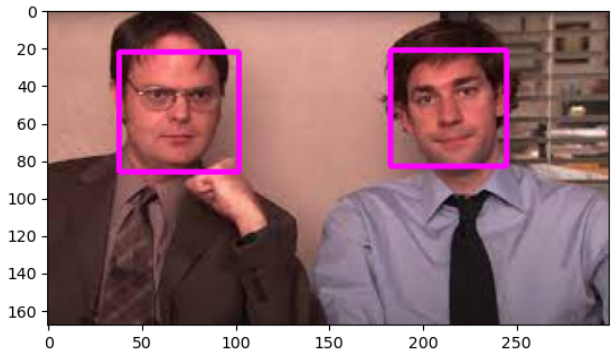
| Epoch | Sparse Categorical Cross-Entropy Loss | Perplexity |
|---|---|---|
| 5 | 2.4374 | 11.4432 |
| 10 | 2.1359 | 8.4646 |
| 15 | 1.9613 | 7.1085 |
| 20 | 1.8444 | 6.3243 |

**LM Validation Perplexities**

| | | |
|---|---|---|
| **GPT2 Tuned:** 15.952 | **GPT2 Untuned:** 30.888 | **GPT-J Untuned:** 20.14 |

Table 3: Training and Accuracy Metrics from Language Model (GPT2 and GPT-J)

Additionally, we demonstrate our model by providing the following input still frame, for which two faces are detected:




This generates the following output:

```
characters:  ['Jim', 'Dwight']
objects:  ['manager', 'office', 'Copier', 'company', 'group']
Jim:  Okay, so Dwight is the new manager of Dunder Mifflin Paper Company.
Dwight Schrute is his old self.  He's been in charge of Scranton for
a long time now, and he's finally starting to show signs of being that
guy he used to be.  I mean, look at him.  Look at how handsome he is
now.
```

Here is another example input, for which only one face is detected:


This generates the following output:

```
characters:  ['Michael']
objects:  ['company', 'desk', 'manager', 'Copier', 'office']
Michael:  Well, that's all I really care about right now.  Okay, well,
let me just tell you what I think you should know about me.  Michael
Scott is a great company manager.  He's been with us for a long time
and he knows exactly what it takes to be a top cop in this company.
So, when it comes down to it, he's exactly the right guy for this job.
But, unfortunately for me, at the end of the day, it just didn't work
out for any of us.  For one thing, Kevin was instrumental in that.  And
for another, Creed is totally my worst enemy.  Which is why I can't
help but think that maybe we should all just fire him.  Because, um...
I mean, maybe Creed
```

# 4 Challenges

Given our architecture of connected, sequential networks, a main challenge of the project was finding compatible data structures and models that could realistically merge into a single pipeline — especially given our time constraint!

- *Compatible Data Structures*: with our need for implementation of data into multiple pre-trained models (e.g. VGG16, ResNet50, GPT2, GPT-J), a challenge was making efficient decisions on initial data preprocessing. The difficulties arose in recognizing a need to predict the input needs of downstream tasks, especially as we combined two separate models into one pipeline. As we learned more and more about the pre-trained models through our implementation process, we found a need to further adapt our data structures to allow for downstream implementation. This re-appearing challenge thus (frustratingly so) consumed a lot of time.

- *Compatible and High-Performing Models*: with our hopes of ensuring high accuracies across all our network components, a challenge of the project was making sure to try out

many different models. Among these tested out models were VGG16, ResNet50, CLIP in PyTorch, CLIP in TensorFlow, GPT2 and GPT-J, all with a number of different implementation methods. Moreover, as these trials were performed, we needed to ensure that all processing methods within models (whether at input, in between, or at output) had to be compatible with upstream and downstream tasks. These needs to deeply research our model options were again very time consuming.

Most of all, the networks we ended up choosing to work with were all conceptually-challenging to research and fully implement. Understanding such novel, powerful models by individually compiling resources was a daunting task, especially as the field is constantly growing and thus leaving behind a trail of deprecated tools that can be difficult to navigate. Thus, we overcame these challenges with rigorous research for resources and new tools, ultimately leading to our final results.

# 5 Reflection

We ultimately feel that our project turned out well, especially given our personal time constraints and workloads. We successfully accomplished each part of the network as planned, with very minor tweaks in choices of models that ultimately still gave us the meaningful output we wanted.

We recall our original goals:

- Base Goal: to generate text resembling a script.
- Target Goal: to generate text that could plausibly have been spoken in the still frame.
- Stretch Goal: to generate text that sounds like a character from The Office in the still frame.

We indeed met our base goal, and we mostly depend on expert knowledge to establish if we met our target and stretch goals. Based on our personal expertise, we believe that much of the output is believably material that mimics the language of the show. We feel quite good about our ability to mostly meet our target and stretch goals.

We feel that our model indeed worked the way we expected it, thanks to much tweaking within network components. In general, our approach stayed consistent over time: we very much followed our game plan of exploring models, finalizing model decisions, and merging into a single pipeline. Given the amount of time that model exploration ended up taking, we did have to trade-off some of our original hopes for our metrics by, for example, prioritizing models that are most compatible for merging. However, we essentially made no pivots from our general plan and goals, so we are quite satisfied with the way we used our time on the project.

If we had more time on the project, there are a few things we would like to improve or look into:

1. *Object Detection Incorporation*: we realize that the identification of objects in still frames ends up playing very heavily into the output dialogue. For example, if `[cat]`

is listed as a noun present in the image, the language model will heavily prioritize the dialogue to include discussions about cats. However, it is clear that there may be objects in scenes that are not the focus of conversation: e.g. there are windows, desks, and mugs in many scenes of *The Office*, but they are not always being discussed. If we had more time, we would try modulating how much objects play into output dialogue. A first step could mean finding a more effective way of generating our objects/nouns list (1), as our current method of using expert knowledge could be introducing unwanted bias.

2. *Limited Data*: our image classification network (2.1) was only trained for character classification on six main characters. This means we are only able to input still frames from the show that contain images of these six characters, whereas our language model network (2.2) can provide dialogue for every character on the show. Thus, if we had a larger dataset of more character images, we could easily train our VGG16 model to classify more characters from the show to maximize the use of our later language model network.

3. *Facial Expression Recognition*: given our preprocessing abilities to zoom and focus on faces in still frames (2.1.1), an extension could be to leverage facial expressions to better understand the sentiments within a scene. This could better inform the generation of dialogue that happens between characters.

4. *More Zero-Shot Results*: many large language models have zero-shot capabilities, but with a lack of time to further explore, we again chose transfer learning on our language model network (2.2). However, it could be interesting to explore how zero-shot results could differ from our current results.

A major takeaway from the project was how important transfer learning can be. Our implementation of powerful models into our architecture was successful thanks only to our research into efficient transfer learning of our models. Additionally, we learned to research and compile resources. Understanding the novel, high-performing models we used required time-consuming compilation of many, many resources. Moreover, learning to navigate through a growing field with deprecated tools taught us adaptibility and persistence.

# References


[1] P. Viola, M. Jones, Rapid Object Detection using a Boosted Cascade of Simple Features (2001).

[2] Wei-Meng Lee, Implementing Face Recognition Using Deep Learning and Support Vector Machines (8 2022).

[3] Q. Cao, L. Shen, W. Xie, O. Parkhi, A. Zisserman, VGGFace2 Dataset.
URL https://www.robots.ox.ac.uk/~vgg/data/vgg_face2/

[4] The Office Characters — Kaggle.
URL https://www.kaggle.com/datasets/pathikghugare/the-office-characters

[5] openai/clip-vit-base-patch32 · Hugging Face.
URL https://huggingface.co/openai/clip-vit-base-patch32

[6] Open AI, Interacting with CLIP.
URL https://github.com/openai/CLIP/blob/main/notebooks/Interacting_with_CLIP.ipynb

[7] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning Transferable Visual Models From Natural Language Supervision.
URL https://github.com/OpenAI/CLIP.

[8] Efficient Training on a Single GPU.
URL https://huggingface.co/docs/transformers/perf_train_gpu_one#tf32

[9] How to fine-tune GPT-2 for beginners — Kaggle.
URL https://www.kaggle.com/code/changyeop/how-to-fine-tune-gpt-2-for-beginners

[10] The Office (US) - Complete Dialogue/Transcript — Kaggle.
URL https://www.kaggle.com/datasets/nasirkhalid24/the-office-us-complete-dialoguetranscript

[11] EleutherAI/gpt-j-6B · Hugging Face.
URL https://huggingface.co/EleutherAI/gpt-j-6B

## 5.1 Additional Image Sources

- https://imagesvc.meredithcorp.io/v3/mm/image?url=https%3A%2F%2Fstatic.onecms.io%2Fwp-content%2Fuploads%2Fsites%2F6%2F2020%2F02%2Foffice-2-2000.jpg
- https://akns-images.eonline.com/eol_images/Entire_Site/2020017/rs_1024x759-200117120856-1024-The-Office-Stress-Relief.jpg?fit=around%7C776:576&output-quality=90&crop=776:576;center,top
- https://helios-i.mashable.com/imagery/articles/03O7JfLqPcz2jP9hdgZspjF/hero-image.fill.size_1200x675.v1632315010.png
- https://www.hollywoodreporter.com/wp-content/uploads/2015/03/the_office_season_1_cast.jpg
- https://www.insider.com/the-office-best-moments-that-were-improvised-on-series
- https://www.floor8.com/posts/video-10-hilarious-the-office-deleted-scenes-01e46m98taw7

- https://screenrant.com/the-office-most-viewed-scenes-youtube/ https://parade.com/1278812/lauren-thoman/best-office-episodes/
- https://tvline.com/2021/01/01/the-office-cold-open-deleted-scene-matrix-series-finale-video/
- https://www.imdb.com/title/tt1535367/
- https://www.cleveland.com/entertainment/2020/03/the-office-at-15-ranking-the-10-best-episodes.html
- https://fandomwire.com/the-office-john-krasinski-american-sitcom-steve-carell/ https://www.nme.com/news/tv/the-office-gas-station-proposal-was-shows-most-expensive-scene-2951008
- https://indianexpress.com/article/entertainment/television/top-10-the-office-episodes-6397126/
- https://www.thepioneerwoman.com/news-entertainment/g34689052/the-office-cast-characters/
- https://www.cheatsheet.com/entertainment/the-office-an-unaired-take-of-jim-seeing-dwight-and-angela-kissing-was-too-much.html/