Pragmatics in Video Games

Alejandro Ciuba, Spring 2022

Google Slides – Commenter Perms, tell me what you think!

NOTE: This is a reformatted copy of my original presentation—all formatting has been changed. This is due to the fact my original layout was a *Slidesgo* layout, which explicitly denies permission to post any work containing their layouts to databases (e.g. *GitHub*). To see a copy of the original (better formatted) presentation, please email me at the gmail address listed in the last slide. Thank you and enjoy!

Overview

The Beginning

The Beginning – Finding The Data

- I had a lot of trouble settling on a research topic
  - So many options, so little time
  - I knew I wanted to do video games, but I had trouble obtaining data

- I almost went a completely different route using ESL data (from *PELIC*)
  - Was still interesting, but not exactly what I was hoping to do

- At the last second, I stumbled upon a GitHub Repository
  - Contained partially cleaned data of various video game texts and dialogues
  - Created by a graduate student: Judith van Stegeren
    - Created for ***Fantastic Strings and Where to Find Them: The Quest for High-Quality Video Game Text Corpora***
  - After a quick email, I had their permission to use the data from their repository

The Beginning – Adding & Cleaning

- Now, I had texts from 3 major video games
  - However, the data was only partially cleaned
  - I wanted more

- Now, to create my own
  - I found a complete script of the game, *Hollow Knight*, as a Google Doc
  - Downloaded an HTML and Plaintext version
  - Created *HDialogueParser.py* to use the HTML tags to tag the headings in the Plaintext version via BeautifulSoup and regular expressions
    - Ended up not needing them, used the tags to remove the headers

- I quickly cleaned the data I now had:
  - This was done in *initial_base_data_exploration.ipynb*
  - Removed NaNs, parsed information from URLs, word counts via NLTK, etc.
  - There’s now more, but we’ll get to those later
  - Most data is publicly available through *.pkls* in my *sample_data* directory

The Beginning – The Data So Far

- At this point, I now had 4 major video game dialogue scripts:

Data Overview

Game Name	Entries	Avg. Token Count	Total Token Count	Additional Comments
TES Books	5,446 Books	351.91 per Book	1,916,513	*Morrowind, Oblivion, Skyrim, TES: Online, Daggerfall, Arena*
*KOTOR**	29,213 Speaker Lines	19.08 per Voiceline	557,261	Came with animations and developer comments
Torchlight II	1008 Quest Texts	33.92 per Text	34,187	Quest Texts not number of in-game quests
Hollow Knight	55 Characters	756.25 per Character	41,594	Context removed – essentially dialogue dump for all characters.

during dialogue in a separate DataFrame

The Beginning – My Original Plan

- What I originally wanted to examine with the data was as follows:
  - **Question I**: How do video game pragmatics and discourse differ from their real life counterparts? Does the player being the central figure drastically affect how they are realized?
  - **Question II**: What (if any) techniques do video game writers employ which are different from other forms of writing?
  - **Question III**: Do writers of fantasy video games take into consideration sociolinguistic elements such as dialect, culture, identity, region, etc. when writing their characters? If so, how?
  - **Question IV**: Expanding upon question 3, are those considerations "realistic"; in other words, do writers commit pseudo-linguistics when trying to write for fantasy cultures?
  - **Question V**: Are there differences between professional video game writers, and "indie" (low-budget, small studio) video game writers?

The Change

The Change – The Many Problems

- I thought the questions were good and would allow for deep research throughout the semester

- Slowly, questions were changed or dropped to match the data better
    - **Question I**: too generalized, no sense of operationalized metrics
    - **Question II**: I only had video game data, comparing it to other forms of writing would mean even more data – it would get out-of-hand
    - **Question III**: dropped this completely, I didn't want to do a sociolinguistic case-study involving only these games – I want to come back to this
    - **Question IV**: again, I didn't feel ready to tackle this or question III
    - **Question V**: way too broad! Also, I know nothing about judging the quality of writing – especially for very different genres

The Change – The *Actual* Questions

- My finalized questions were:
  - **Question I**: How are orders/requests realized in video game dialogues? Are there more direct or indirect orders?
  - **Question II**: What is the frequency of the 2nd person pronoun, you?
    - Extending that, what are the frequencies of other pronouns?
  - **Question III**: Are hapaxes a common occurrence in video games?
    - Extending this, what named entity types are common?

- I finally produced a set of questions that I felt I could really dig into
  - These questions, unlike the original, were much more exploratory than anything
  - Focused on the data I had and what conclusions (if any) I could make from them
  - Much more manageable and more care could be put into each step

The Change – The New Plan

- I now had goals; however, there was still a lot of work to be done
  - How would I answer these questions?
  - What tools would I need?
  - In what format(s) my data would need to be?

- Let’s examine it on a question-by-question basis:
  - **Question I**:
    - Final Goal: a list of clauses that are (mostly) orders and requests
    - Tasks: create a list of common order/request clausal/sent. forms, figure out a way (or ways) to systematically capture them
  - **Question II**:
    - Final Goal: percentages of pronouns per video game
    - Tasks: again, how to capture them?
  - **Question III**:
    - Final Goal: a list of named entities and hapaxes
    - Tasks: once again, how to capture them *and* how to store them?

Introducing *spaCy*

Introducing *spaCy* – Brief Overview

- What is *spaCy?*
  - An NLP pipeline
  - Essentially, you download a given ML model for your needs and *spaCy* will use that model to:
    - Create a "Doc" Object: an object containing all parsed information on the fed-in text
    - Tokenized words with various linguistic information: lemma, part-of-speech (universal), tag (pennTree-style), pred. arg. role, word-vector (transformer models only), morphological info., dependencies (left and right), IOB (named-entity recognition scheme)
    - Tokenized sentences containing dependency trees
  - Then, you can use this information to parse through the text similarly to regular expressions!
    - Useful tools for token-matching, phrase-matching, entity-ruling

- **THIS WAS EXTREMELY POWERFUL AND FUN TO USE!!!**

Introducing *spaCy* – Even More Data!

- I created a couple plaintext files containing all the dialogue of *Torchlight II* and *Hollow Knight* – essentially a data dump of each DataFrame's *text* column!

- I also created plaintext files of randomly-sampled data from *TES Books* and *KOTOR*
  - There's a token limit to *spaCy* Doc Objects since they take up *a lot* of space!

- Created 4 Doc Objects, the same model running through each text
  - Pickled them for future use, because training took about 20 minutes originally

- I now had a lot of linguistic information I could use for various purposes!
  - For example, I could use both regex and *spaCy's Matcher* to find certain sentence structures!

Orders & Requests

Orders & Requests – The Process

- I now had enough information to really capture sentence structures I was looking for!

- I sat down and wrote out some generalized common ways one can phrase orders and requests:
  - For example: *Mand. Form, I/we order/demand/force/command (that) you…, May/Would/Can/Could you…*
  - There were 13 different forms I wrote down and generalized
  - Note: we will get false negatives and positives because we lose a lot of *utterance context!*
    - I also had to exclude forms because of this (e.g. *"Your homework isn't done!"* can be an indirect request or direct assertion)

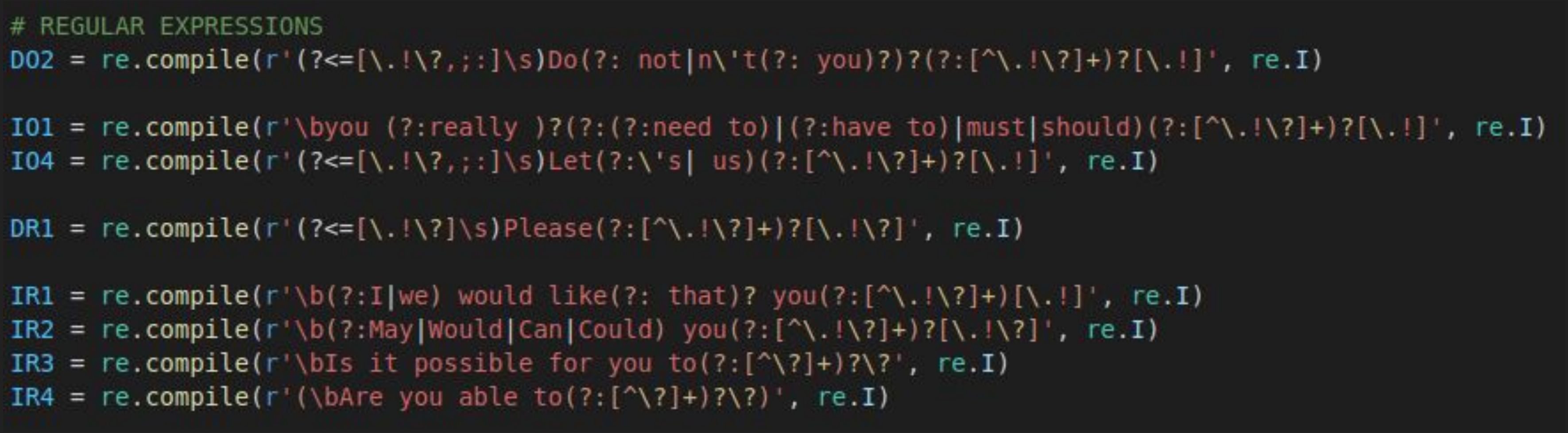
Orders & Requests – The Structures

- Direct Orders:
  - *Mand. Form*
  - *Don't/Do not/Do Mand. Form X.*
  - *I/we order/demand/force/command (that) you X.* **(Performative Speech Act)**
- Indirect Orders:
  - *You /need to/have to/must/should X.*
  - *Your objective/goal/quest/task is X.*
  - *I/we need/want/require/desire you Y.*
  - *Let's/Let us X.*

- Direct Requests:
  - *Please X.*
  - *I/we request (that you)/ask (of you) X.* **(Performative Speech Act)**
- Indirect Requests:
  - *I/we would like (that) you Y.*
  - *May/Would/Can/Could you X?*
  - *Is it possible for you to X?*
  - *Are you able to X?*

Orders & Requests – The Regex

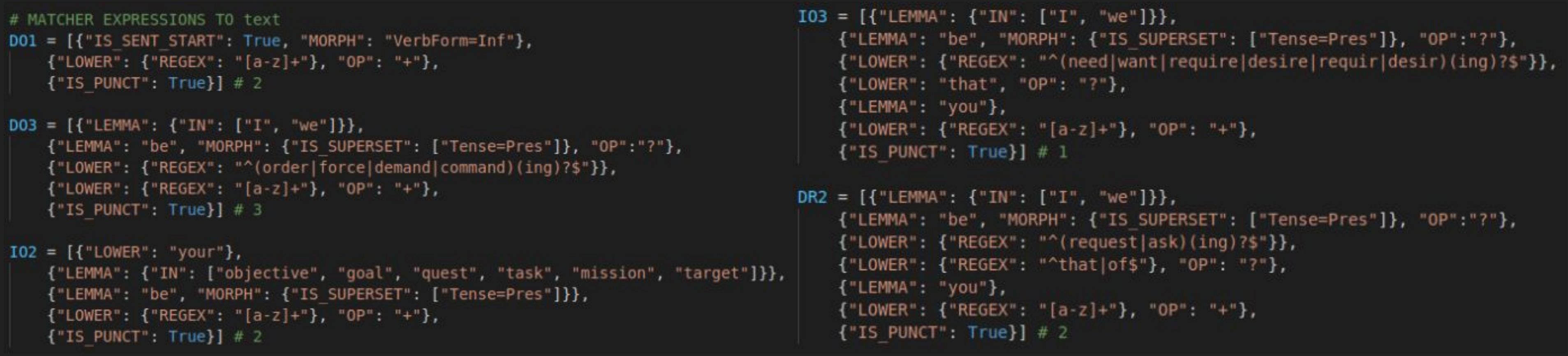
- I didn’t use *spaCy’s Matcher* for everything – memory efficiency and time!
  - **If I could more easily capture a structure with good ol’ fashioned regex, I would!**
- Regex consist of a head (prereqs), a body (struct-to-capture), and a footer (full-sent)



```python
# REGULAR EXPRESSIONS
DO2 = re.compile(r'(?<=[\.!\?,;:]\s)Do(?: not|n\'t(?: you)?)?(?:[^\.!\?]+)?[\.!]', re.I)

IO1 = re.compile(r'\byou (?:really )?(?:(?:need to)|(?:have to)|must|should)(?:[^\.!\?]+)?[\.!]', re.I)
IO4 = re.compile(r'(?<=[\.!\?,;:]\s)Let(?:\'s| us)(?:[^\.!\?]+)?[\.!]', re.I)

DR1 = re.compile(r'(?<=[\.!\?]\s)Please(?:[^\.!\?]+)?[\.!\?]', re.I)

IR1 = re.compile(r'\b(?:I|we) would like(?: that)? you(?:[^\.!\?]+)[\.!]', re.I)
IR2 = re.compile(r'\b(?:May|Would|Can|Could) you(?:[^\.!\?]+)?[\.!\?]', re.I)
IR3 = re.compile(r'\bIs it possible for you to(?:[^\?]+)?\?', re.I)
IR4 = re.compile(r'(\bAre you able to(?:[^\?]+)?\?)', re.I)
```

Orders & Requests – The Patterns

- While powerful, the *Matcher* only goes on a per-Token basis – I couldn’t look-ahead or look-behind sometimes!
  - However, some structures definitely needed either lemma or morphological info.
  - Some were easier to think about with the *Matcher’s* JSON format

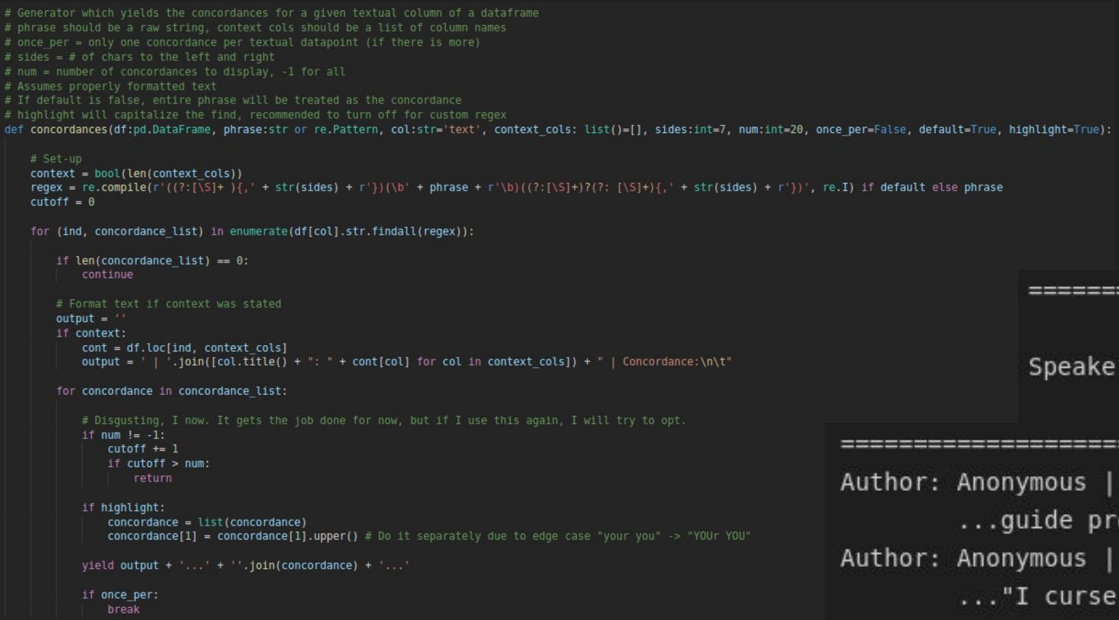


```
# MATCHER EXPRESSIONS TO text
DO1 = [{"IS_SENT_START": True, "MORPH": "VerbForm=Inf"},
    {"LOWER": {"REGEX": "[a-z]+"}, "OP": "+"},
    {"IS_PUNCT": True}] # 2

DO3 = [{"LEMMA": {"IN": ["I", "we"]}},
    {"LEMMA": "be", "MORPH": {"IS_SUPERSET": ["Tense=Pres"]}, "OP":"?"},
    {"LOWER": {"REGEX": "^(order|force|demand|command)(ing)?$"}},
    {"LOWER": {"REGEX": "[a-z]+"}, "OP": "+"},
    {"IS_PUNCT": True}] # 3

IO2 = [{"LOWER": "your"},
    {"LEMMA": {"IN": ["objective", "goal", "quest", "task", "mission", "target"]}},
    {"LEMMA": "be", "MORPH": {"IS_SUPERSET": ["Tense=Pres"]}},
    {"LOWER": {"REGEX": "[a-z]+"}, "OP": "+"},
    {"IS_PUNCT": True}] # 2
```

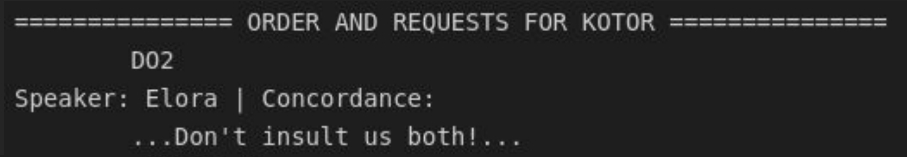
```
IO3 = [{"LEMMA": {"IN": ["I", "we"]}},
    {"LEMMA": "be", "MORPH": {"IS_SUPERSET": ["Tense=Pres"]}, "OP":"?"},
    {"LOWER": {"REGEX": "^(need|want|require|desire|requir|desir)(ing)?$"}},
    {"LOWER": "that", "OP": "?"},
    {"LEMMA": "you"},
    {"LOWER": {"REGEX": "[a-z]+"}, "OP": "+"},
    {"IS_PUNCT": True}] # 1

DR2 = [{"LEMMA": {"IN": ["I", "we"]}},
    {"LEMMA": "be", "MORPH": {"IS_SUPERSET": ["Tense=Pres"]}, "OP":"?"},
    {"LOWER": {"REGEX": "^(request|ask)(ing)?$"}},
    {"LOWER": {"REGEX": "^that|of$"}, "OP": "?"},
    {"LEMMA": "you"},
    {"LOWER": {"REGEX": "[a-z]+"}, "OP": "+"},
    {"IS_PUNCT": True}] # 2
```

Orders & Requests – One Last Thing!

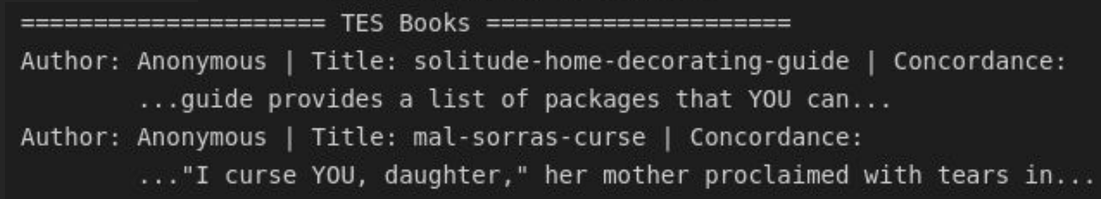
- I also needed to capture both these structures and concordances (for pronouns later)
  - **I made an extremely useful concordance generator – memory meh-efficient**



```python
# Generator which yields the concordances for a given textual column of a dataframe
# phrase should be a raw string, context_cols should be a list of column names
# once_per = only one concordance per textual datapoint (if there is more)
# sides = # of chars to the left and right
# num = number of concordances to display, -1 for all
# Assumes properly formatted text
# If default is false, entire phrase will be treated as the concordance
# highlight will capitalize the find, recommended to turn off for custom regex
def concordances(df:pd.DataFrame, phrase:str or re.Pattern, col:str='text', context_cols: list()=[], sides:int=7, num:int=20, once_per=False, default=True, highlight=True):

    # Set-up
    context = bool(len(context_cols))
    regex = re.compile(r'((?:[\S]+ ){,' + str(sides) + r'})(\b' + phrase + r'\b)((?:[\S]+)?(?: [\S]+){,' + str(sides) + r'})', re.I) if default else phrase
    cutoff = 0

    for (ind, concordance_list) in enumerate(df[col].str.findall(regex)):

        if len(concordance_list) == 0:
            continue

        # Format text if context was stated
        output = ''
        if context:
            cont = df.loc[ind, context_cols]
            output = ' | '.join([col.title() + ": " + cont[col] for col in context_cols]) + " | Concordance:\n\t"

        for concordance in concordance_list:

            # Disgusting, I now. It gets the job done for now, but if I use this again, I will try to opt.
            if num != -1:
                cutoff += 1
                if cutoff > num:
                    return

            if highlight:
                concordance = list(concordance)
                concordance[1] = concordance[1].upper() # Do it separately due to edge case "your you" -> "YOUr YOU"

            yield output + '...' + ''.join(concordance) + '...'

            if once_per:
                break
```



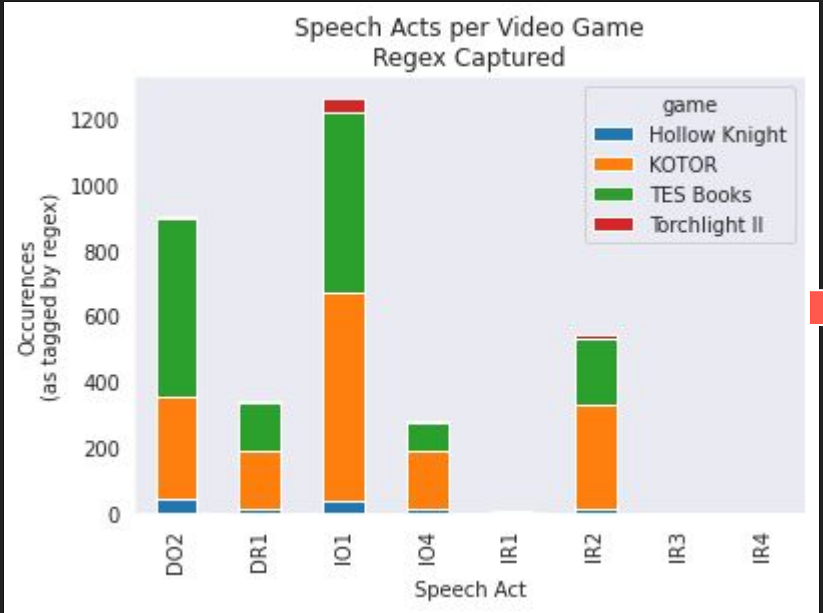
```
=============== ORDER AND REQUESTS FOR KOTOR ===============
        D02
Speaker: Elora | Concordance:
        ...Don't insult us both!...
```

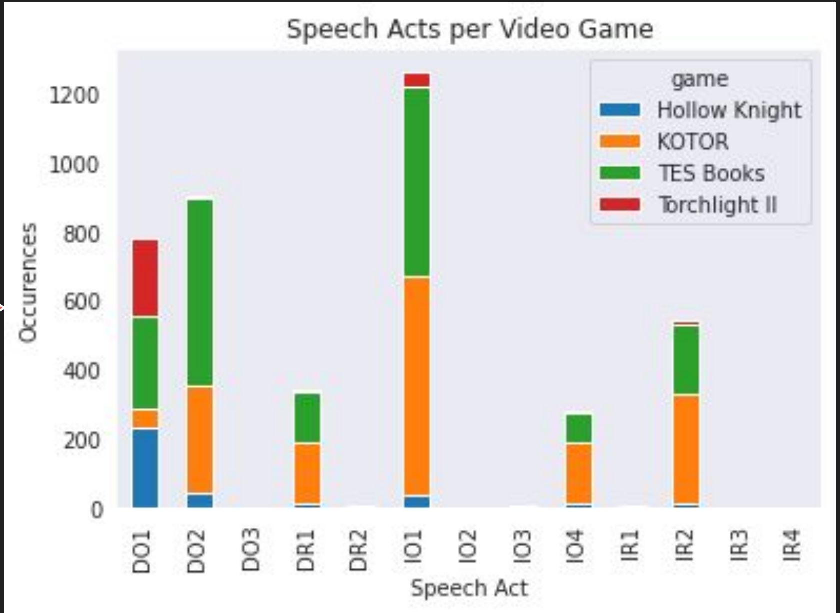


```
===================== TES Books =====================
Author: Anonymous | Title: solitude-home-decorating-guide | Concordance:
        ...guide provides a list of packages that YOU can...
Author: Anonymous | Title: mal-sorras-curse | Concordance:
        ..."I curse YOU, daughter," her mother proclaimed with tears in...
```

Orders & Requests – The Results

- My patterns and regex captured things quite well – overall, even false positives are easily detected upon examination and are very infrequent
    - **False negatives just don't show up! So there's no way to know the recall rate…**



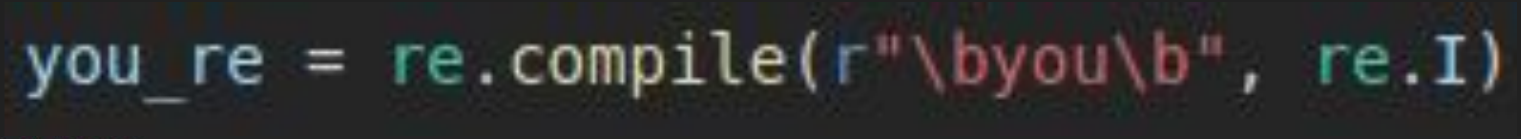


Pronoun Frequencies

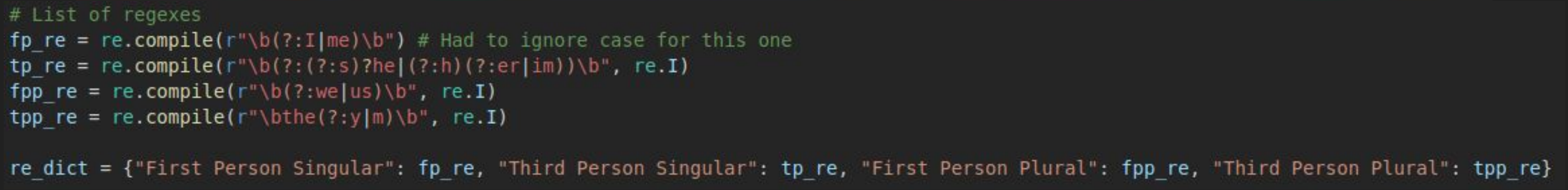
Pronoun Freq. – The Process

- Way more straightforward than orders & requests
  - Literally just compile a bunch of pretty simple regex
  - My concordances function was originally written for this part
  - First thing I tackled

- I got concordances for *you* and also graphed percentages for all pronouns I wanted



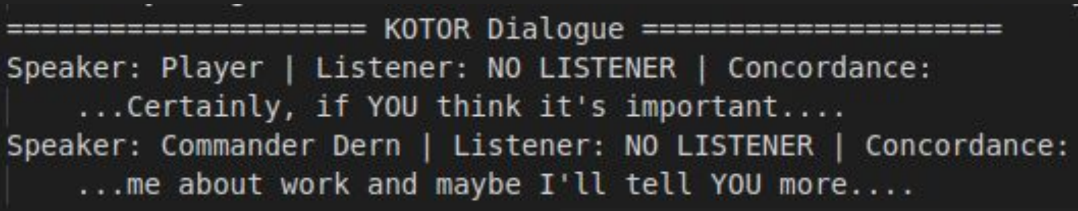
```
you_re = re.compile(r"\byou\b", re.I)
```



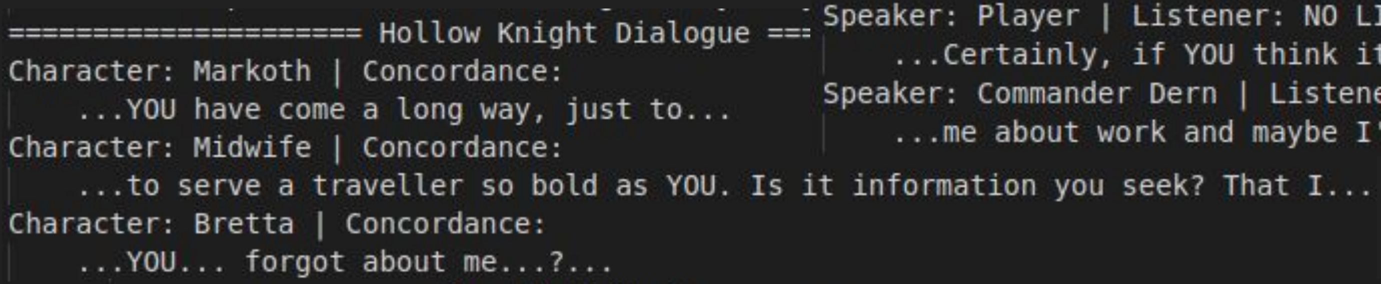
```
# List of regexes
fp_re = re.compile(r"\b(?:I|me)\b") # Had to ignore case for this one
tp_re = re.compile(r"\b(?:(?:s)?he|(?:h)(?:er|im))\b", re.I)
fpp_re = re.compile(r"\b(?:we|us)\b", re.I)
tpp_re = re.compile(r"\bthe(?:y|m)\b", re.I)

re_dict = {"First Person Singular": fp_re, "Third Person Singular": tp_re, "First Person Plural": fpp_re, "Third Person Plural": tpp_re}
```

Pronoun Freq. – Samples In-Use

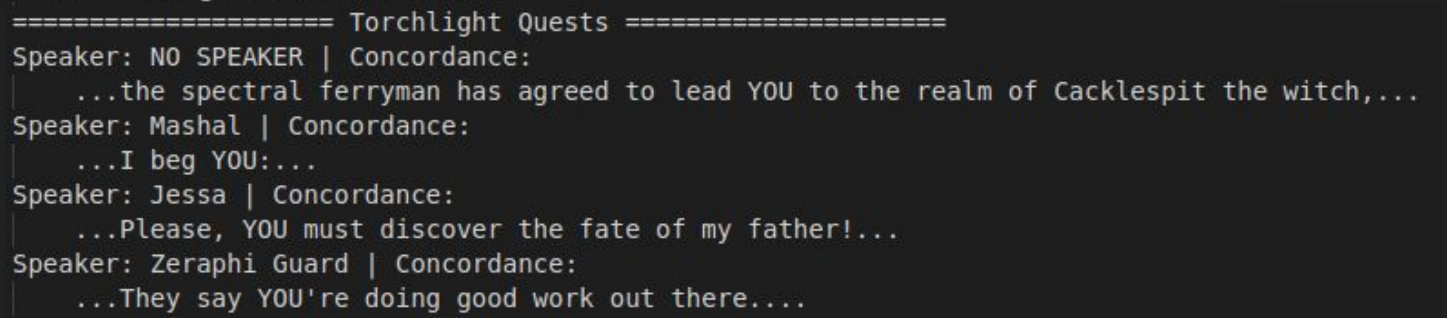
- There’s a bigger sample in the *research.ipynb* notebook, here’s just a few samples
  - I think these regexes are pretty accurate and don’t miss much (if anything!)



```
====================== KOTOR Dialogue ======================
Speaker: Player | Listener: NO LISTENER | Concordance:
    ...Certainly, if YOU think it's important....
Speaker: Commander Dern | Listener: NO LISTENER | Concordance:
    ...me about work and maybe I'll tell YOU more....
```



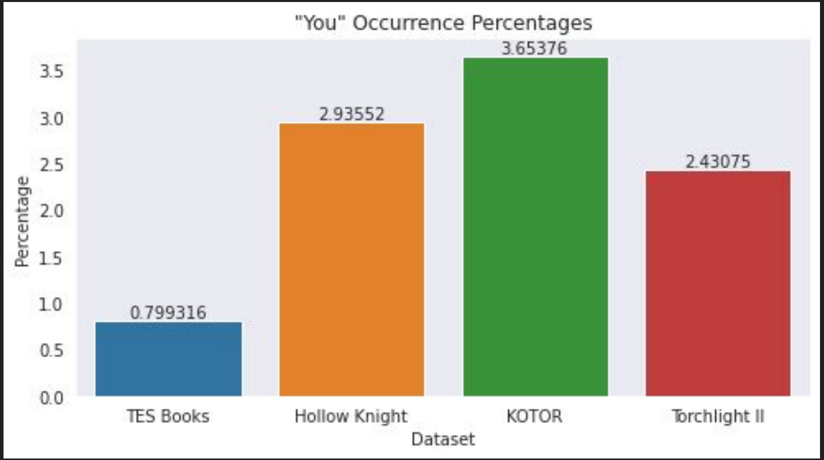
```
===================== Hollow Knight Dialogue ===
Character: Markoth | Concordance:
    ...YOU have come a long way, just to...
Character: Midwife | Concordance:
    ...to serve a traveller so bold as YOU. Is it information you seek? That I...
Character: Bretta | Concordance:
    ...YOU... forgot about me...?...
```

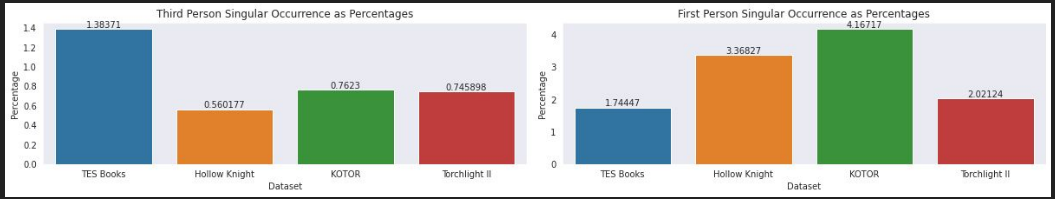


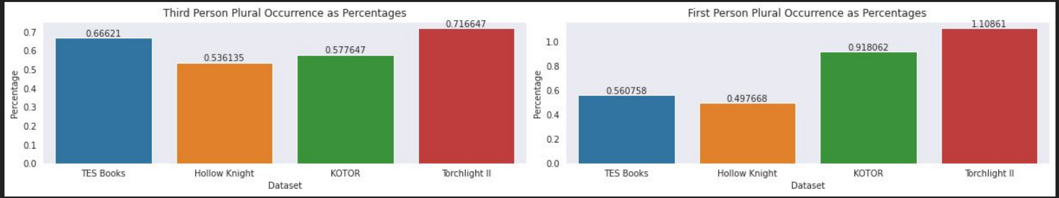
```
==================== Torchlight Quests ====================
Speaker: NO SPEAKER | Concordance:
    ...the spectral ferryman has agreed to lead YOU to the realm of Cacklespit the witch,...
Speaker: Mashal | Concordance:
    ...I beg YOU:...
Speaker: Jessa | Concordance:
    ...Please, YOU must discover the fate of my father!...
Speaker: Zeraphi Guard | Concordance:
    ...They say YOU're doing good work out there....
```

Pronoun Freq. – The Results

- Below are the percentage of tokens each pronoun/pronoun group made up for each game
  - I ignored possessive pronouns and “y’all”
  - Results *definitely* reflect the type of data from each game







Named Entities

Named Entities – The Process

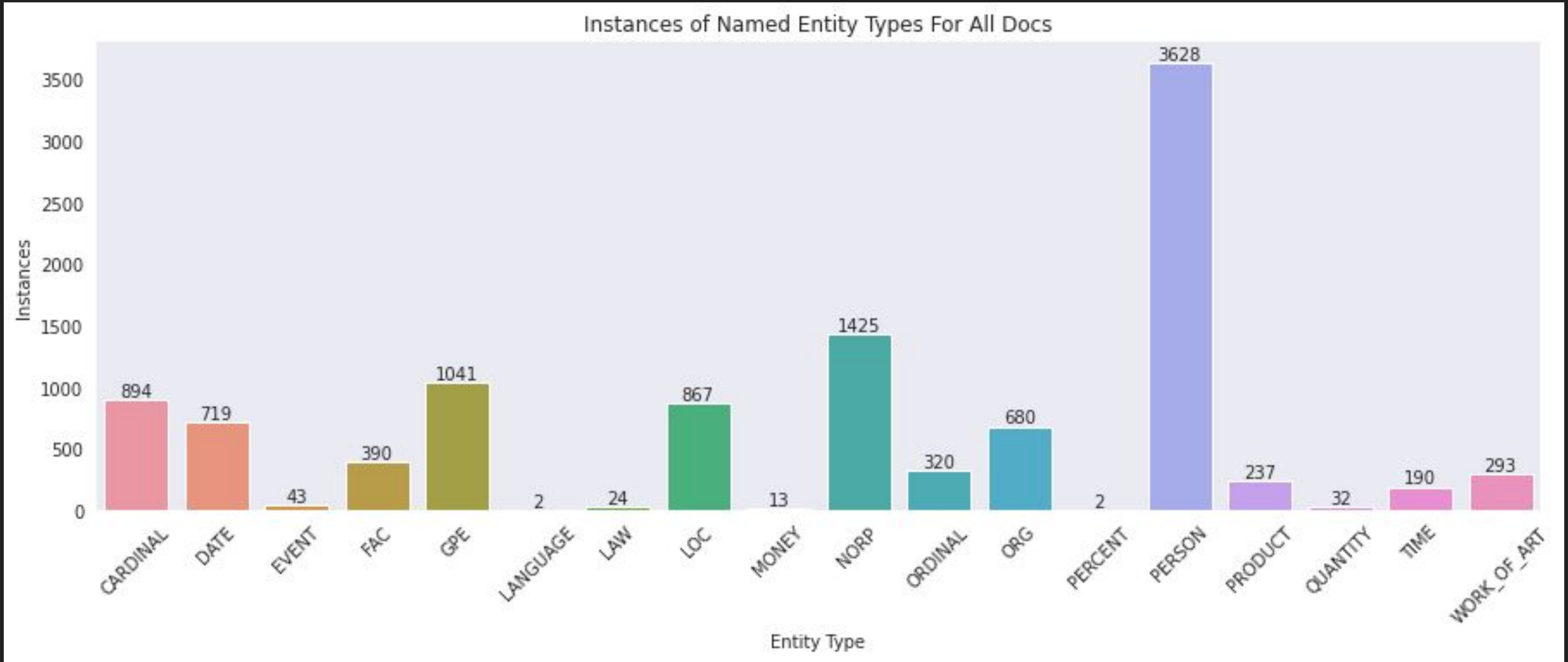
- When creating a Doc Object, *spaCy* automatically tags named entities and stores then in the Doc’s *.ents* field
  - Creating the DataFrame was as simple as iterating through each doc object

- Now, I had a DataFrame that contained information about each named entity
  - *entities.pkl* in the *sample_data* subdirectory

- While good, I noticed it wasn’t great, even using the transformer model
  - Certain entries are… questionable
    - Toponyms are very problematic
  - However, the results seemed good enough to get some good data
    - Might refine later… probably not

Named Entities – The Results

- I don’t know why the chart turned out so different from all the others, but here it is!

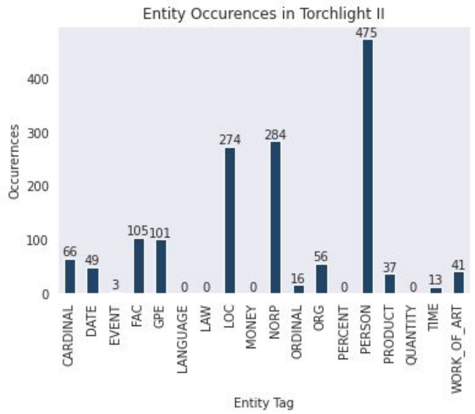


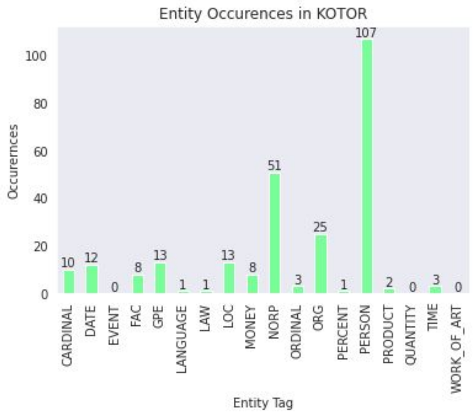
Named Entities – The Results II

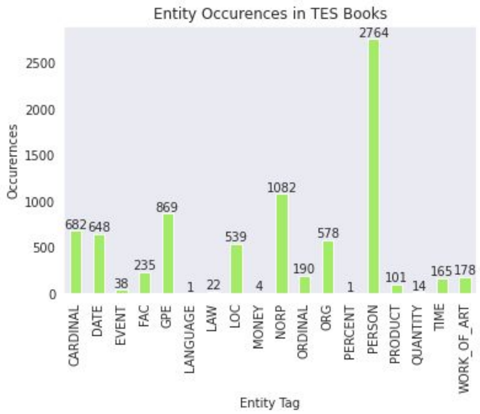
- Same results, but on a "per game" basis!

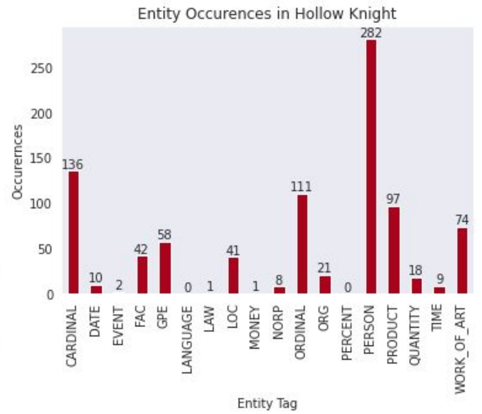
- As you can see, *PERSON* is by far the most prevalent
  - However, I am unsure as a smaller model produced different results
  - I also still believe *ORG* or *NORP* should be the most popular

- Surprisingly, there's low amounts of *TIME* and *MONEY* labels too!



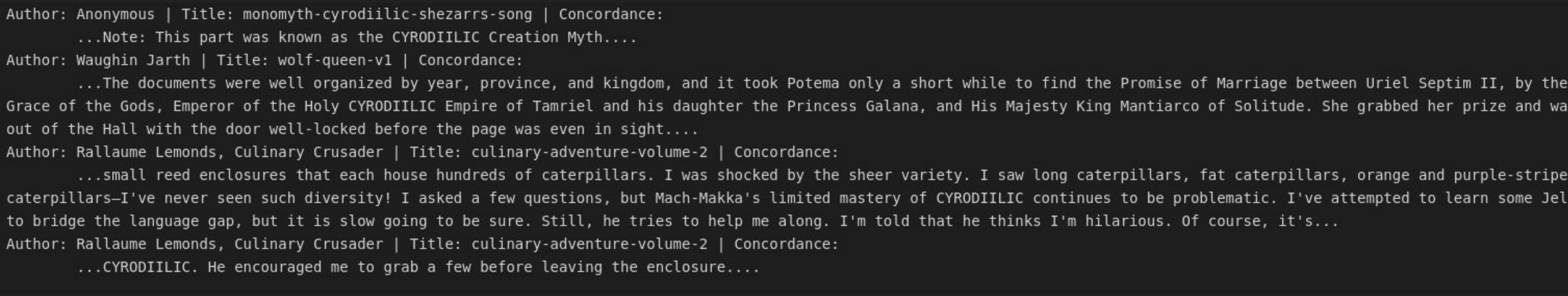






Named Entities – In-Use

- Below are the results of the named-entity *Cyrodiilic* (a *LANG*):
    - As you can see, *toponyms* are a problem for my data



```
Author: Anonymous | Title: monomyth-cyrodiilic-shezarrs-song | Concordance:
        ...Note: This part was known as the CYRODIILIC Creation Myth....
Author: Waughin Jarth | Title: wolf-queen-v1 | Concordance:
        ...The documents were well organized by year, province, and kingdom, and it took Potema only a short while to find the Promise of Marriage between Uriel Septim II, by the
Grace of the Gods, Emperor of the Holy CYRODIILIC Empire of Tamriel and his daughter the Princess Galana, and His Majesty King Mantiarco of Solitude. She grabbed her prize and wa
out of the Hall with the door well-locked before the page was even in sight....
Author: Rallaume Lemonds, Culinary Crusader | Title: culinary-adventure-volume-2 | Concordance:
        ...small reed enclosures that each house hundreds of caterpillars. I was shocked by the sheer variety. I saw long caterpillars, fat caterpillars, orange and purple-stripe
caterpillars—I've never seen such diversity! I asked a few questions, but Mach-Makka's limited mastery of CYRODIILIC continues to be problematic. I've attempted to learn some Jel
to bridge the language gap, but it is slow going to be sure. Still, he tries to help me along. I'm told that he thinks I'm hilarious. Of course, it's...
Author: Rallaume Lemonds, Culinary Crusader | Title: culinary-adventure-volume-2 | Concordance:
        ...CYRODIILIC. He encouraged me to grab a few before leaving the enclosure....
```

Named Entities – Hapaxes

- Well… I'm not done with them yet
  - I am unsure if I will even include them in the final version
- While I can get the hapaxes from *Hollow Knight* and *Torchlight II* very easily, the other two pose a problem
  - Remember how the other two were only samples?
  - Running my concordance function on them highlights that they aren't *actually* hapaxes – biggest offender "Hutt" from *KOTOR*
- I still want to try to do it, as I think it could be interesting.
  - Most of *Hollow Knight's* hapaxes are numbers, I haven't looked at *Torchlight II's* hapaxes yet, but I assume a similar trend
  - I expected a lot from *TES Books* (even in samples), but I forgot that they consistently use their fantasy vocabulary

Bibliography

- Dataset Based On:
  - van Stegeren, J., & Theune, M. (2020). Fantastic Strings and Where to Find Them: The Quest for High-Quality Video Game Text Corpora. In Intelligent Narrative Technologies Workshop. essay, AAAI Press.
- Data Collected From:
  - Bioware. (2003). Star Wars: Knights of the Old Republic (PC Version) [Video Game]. LucasArts.
  - Torchlight II (PC Version) [Video Game]. (2012). Runic Games.
  - The Elder Scrolls I-V and The Elder Scrolls Online (PC Versions) [Video Games]. (1994-2014). Bethesda Softworks.
  - Hollow Knight (PC Version) [Video Game]. (2017). Team Cherry.

Thanks!

alejandrociuba@pitt.edu/gmail.com
https://alejandrociuba.github.io