Lecture 5: Project Proposals

Padhraic Smyth
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine
Announcements

• **Office hours:**
  – Instructor: Thursdays, 4 to 5:30
    • See online Google signup sheet (link available in Ed)
  – Sakshi (TA): Fridays 10 to 11 and Mondays 10 to 11

• **Discussion sections** with TA Sakshi: Thursday 1 to 2, 2 to 3
  – students can discuss project interests, teaming up: moderated by TA

• **EdD Discussion Board**
  – Post questions about possible projects, questions from lectures, etc

• **Project Proposal**
  – Due Tuesday night next week
## CS 175 Winter 2022 Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Monday</th>
<th>Wednesday</th>
<th>Student Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 3</td>
<td>Lecture: Introduction; class projects</td>
<td>Lecture: Text Classification 1</td>
<td>Work on Assignment 1</td>
</tr>
<tr>
<td>Jan 10</td>
<td>Lecture: Text Classification 2&lt;br&gt;<strong>Assignment 1 due, Tuesday 11:59pm</strong></td>
<td>Lecture: Neural Network Models 1</td>
<td>Work on Assignment 2</td>
</tr>
<tr>
<td>Jan 17</td>
<td>No class (university holiday)&lt;br&gt;<strong>Assignment 2 due, Tuesday 11:59pm</strong></td>
<td><strong>Lecture: Project Proposals</strong></td>
<td>Form teams; work on project proposal</td>
</tr>
<tr>
<td>Jan 24</td>
<td>Lecture: More on Neural Networks&lt;br&gt;<strong>Project proposal due, Tuesday 11:59pm</strong></td>
<td>Lecture: Evaluation Methodologies</td>
<td>Submit proposal; Begin project</td>
</tr>
<tr>
<td>Jan 31</td>
<td>Office hours (no lecture)</td>
<td>Office hours (no lecture)</td>
<td>Work on project</td>
</tr>
<tr>
<td>Feb 7</td>
<td>Office hours (no lecture)</td>
<td>Office hours (no lecture)</td>
<td>Work on project</td>
</tr>
<tr>
<td>Feb 14</td>
<td>No class (university holiday)</td>
<td>Short lecture: Discussion of progress reports&lt;br&gt;<strong>Progress report due, Sunday 11:59pm</strong></td>
<td>Work on project; write progress report</td>
</tr>
<tr>
<td>Feb 21</td>
<td>Project Presentations (in class)&lt;br&gt;<strong>Upload material by 11:59pm Sunday</strong></td>
<td>Project Presentations (in class)&lt;br&gt;<strong>Upload material by 11:59pm Tuesday</strong></td>
<td>Work on project; make short project presentation</td>
</tr>
<tr>
<td>Feb 28</td>
<td>Office hours (no lecture)</td>
<td>Office hours (no lecture)</td>
<td>Work on project</td>
</tr>
<tr>
<td>Mar 7</td>
<td>Short lecture: Discuss final reports</td>
<td>Office hours (no lecture)</td>
<td>Finish project, write final report</td>
</tr>
<tr>
<td>Mar 14</td>
<td><strong>Final project reports due Monday 9am</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Course Grading

<table>
<thead>
<tr>
<th>Activity</th>
<th>Grade Percentage</th>
<th>Date Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment 1</td>
<td>10%</td>
<td>Tuesday Jan 11th</td>
</tr>
<tr>
<td>Assignment 2</td>
<td>10%</td>
<td>Tuesday Jan 18th</td>
</tr>
<tr>
<td>Project Proposal</td>
<td>20%</td>
<td>Tuesday Jan 24th</td>
</tr>
<tr>
<td>Progress Report</td>
<td>20%</td>
<td>Sunday Feb 20th</td>
</tr>
<tr>
<td>Weekly Logs</td>
<td>10%</td>
<td>Weeks 4 to 10</td>
</tr>
<tr>
<td>Presentation/Demo</td>
<td>5%</td>
<td>Feb 21/23</td>
</tr>
<tr>
<td>Final Report</td>
<td>25%</td>
<td>Monday Mar 14th</td>
</tr>
</tbody>
</table>
Project Proposals

• Due Tuesday night next week, worth 20% of your grade

• Find a project team
  – Use Google Sheet, reach out to team members, go to discussion section Thursday

• Decide on a project topic with your team

• Review instructions for proposals

• Write your proposal
  (we will review the structure of proposals later in this lecture)
General Rules for Projects

• **3 students per project**
  – Submit only **one project proposal** per team
    • include all team member names and IDs on the proposal
  – All team members will get the same grade for the proposal, progress report and final report

• **Use of external code is allowed and encouraged**
  – Such code needs to be acknowledged in your reports

• **You must write at least some functionality on your own**
  – As a computer science project, you obviously will need to write your own code too
  – What you implement is up to you
  – You will need to submit the code at the end of the quarter
    (we might not run it but we will look at it)
Finding Project Teams

• Make sure your information is in the Google Sheet

• If you are a 3-person team already:
  – Make sure everyone’s name is associated with the team

• If you are 2 students looking for a 3rd
  – Find individuals with similar interests in the Google Sheet and email them

• If you are 1 student looking for 2 more
  – Find people with similar interests in the Google Sheet and email them
  – It’s fine to find just 1 person initially, and then search for a 3rd
  – Feel free to contact any 2-person teams in the Sheet

• Both individuals and 2-student groups
  – Attend Discussion Section on Thursday to find candidate team members

IMPORTANT: please update the sheet every time your individual or team status changes. This is important so that we can track who is on a team and who is not.
Planning a Project

- **Topic**: select a general type of problem you are interested in, e.g.,
  - Classification, information extraction, summarization, synthesis, parsing, etc

- **Do some background reading to learn more about the topic**
  - Please take a look at the resources on the course Website

- **Define your task + technical approach with your team members**

- **Determine at least 1 or 2 data sets you can use for your project**

- **Discuss how you will evaluate your results, e.g., to compare A v B**
  - Experiments: classification accuracy, precision/recall, etc
  - User Studies: human users compare results from A and B
Questions from Office Hours

Question:
*We need to develop a crawler and parser to gather data for our project (e.g., crawling Web sites for resumes and job ads). Is the work on developing a crawler and parser be considered part of our project?*

Answer:

Yes, this can be counted as part of your project work.

But it should not take too large a fraction of your time (e.g., 1 week or so at most)

Otherwise it means you are spending too much time just on collecting data rather than developing algorithms.
Questions from Office Hours

Question:
We are working on a challenging text classification problem where there are 10 different classes (e.g., different types of sentiment, different genres of movies or music, etc). From reading research papers on this topic that it can be difficult classify data into all 10 classes accurately. What do you suggest?

Answer:
It is fine (particularly when starting out) to select a subset of the easier classes to work with, rather than all of the classes. For example, in movies you could pick Horror versus Romance, and ignore all other classes.

This will allow you to test your classifier on a problem where you expect to be able to accuracy better than random

You can then work up to more complex problems, e.g.,

- 3 classes, e.g., the 2 original classes plus a 3rd class consisting of all of the other classes
Questions?
Suggestions for Project Topics
General Tips for Finding Project Ideas

- Relevant links on course Web page
  https://www.ics.uci.edu/~smyth/courses/cs175/project_reading.html

- Read chapters in Jurafsky and Martin text
  https://web.stanford.edu/~jurafsky/slp3/

- Look at research datasets and associated benchmark problems
  http://nlpprogress.com/
  https://paperswithcode.com/area/natural-language-processing
  https://www.ics.uci.edu/~smyth/courses/cs175/text_data_sets.html

- Google Scholar search on topics of interest, e.g., [twitter sentiment]

- Review of past project reports
  (examples will be posted on Ed.....soon)
Terminology (not yet discussed in lectures)

- **Word embeddings**
  - feature vectors for individual words, e.g., “dog” -> [0.52, -3.11, 4.71, -9.22]
  - Very useful and widely used in NLP

- **Transformed Language Models**
  - Current state-of-the-art for many NLP tasks
  - Use a technique called “attention” (different to RNNs)
  - Have various names: ELMO, BERT, GPT-2/GPT-3, Grover, ...
  - You can use simpler versions in projects, e.g., DistillBERT
  - Come with a cost: quadratic in length of sequence for training and prediction
Terminology (not yet discussed in lectures)

- **Word embeddings**
  - feature vectors for individual words, e.g., “dog” -> [0.52, -3.11, 4.71, -9.22]
  - Very useful and widely used in NLP

- **Transformed Language Models**
  - Current state-of-the-art for many NLP tasks
  - Use a technique called “attention” (different to RNNs)
  - Have various names: ELMO, BERT, GPT-2/GPT-3, Grover, ...
  - You can use simpler versions in projects, e.g., DistillBERT
  - Come with a cost: quadratic in length of sequence for training and prediction

- **Pretrained embeddings and models**
  - Embeddings and models that have pretrained on very large corpora (e.g., by Google)
  - Freely available for students and researchers to use (can be downloaded)

- **Fine-tuning, transfer learning**
  - Taking a large pretrained model and fine-tuning the last layer or two on a specific task
  - Also can use “prompts” (no fine-tuning, just design clever inputs to a language model)
# Examples of Types of Projects

<table>
<thead>
<tr>
<th>Type</th>
<th>Goal</th>
<th>Examples of Data Sets</th>
<th>Evaluation Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Predict sentiment (pos/neg) or stars from text</td>
<td>Twitter, Yelp, Product or Movie Reviews</td>
<td>Classification accuracy, Precision-recall, ...</td>
</tr>
<tr>
<td>Text Summarization</td>
<td>Summarize a set of documents</td>
<td>Product or Movie Reviews</td>
<td>BLEU scores, User studies</td>
</tr>
<tr>
<td>Information Extraction</td>
<td>Product or restaurant reviews</td>
<td>Product Reviews</td>
<td>Accuracy (if labeled), otherwise user studies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>News Articles</td>
<td></td>
</tr>
<tr>
<td>Text Generation</td>
<td>Generate new text in the style of an author</td>
<td>Articles/books/songs by different authors</td>
<td>User studies</td>
</tr>
<tr>
<td></td>
<td>Generate an appropriate response in a dialog (chatbot)</td>
<td>Dialog transcripts</td>
<td></td>
</tr>
<tr>
<td>Question Answering</td>
<td>Generate an answer to a question</td>
<td>Q&amp;A data sets for research projects</td>
<td>Accuracy, precision-recall</td>
</tr>
</tbody>
</table>

See [https://www.ics.uci.edu/~smyth/courses/cs175/project_reading.html](https://www.ics.uci.edu/~smyth/courses/cs175/project_reading.html) for more examples and details.
# Examples: CS 175 Projects in Winter 2021

<table>
<thead>
<tr>
<th>Topic/Task</th>
<th>Methods/Algorithms</th>
<th>DataSets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chatbot about sports</td>
<td>Encoder-decoder</td>
<td>Reddit</td>
</tr>
<tr>
<td>Chatbot</td>
<td>Encoder-decoder+attention</td>
<td>Multiple</td>
</tr>
<tr>
<td>Chatbot</td>
<td>IR-based chatbot</td>
<td>AmazonTopicalChat</td>
</tr>
<tr>
<td>Translate Biblical text</td>
<td>LSTM and transformers</td>
<td>Bible Corpus</td>
</tr>
<tr>
<td>Sentiment prediction</td>
<td>Various classifiers</td>
<td>IMDB/SST/Yelp</td>
</tr>
<tr>
<td>Sentiment prediction</td>
<td>CNN and RNN classifiers</td>
<td>Amazon Reviews</td>
</tr>
<tr>
<td>Classify toxic comments</td>
<td>Various classifiers</td>
<td>Kaggle Toxic Comments</td>
</tr>
<tr>
<td>Story generation</td>
<td>RNNs/LSTMs</td>
<td>Gutenberg books</td>
</tr>
<tr>
<td>Spam detection</td>
<td>Standard classifiers</td>
<td>SMS spam dataset</td>
</tr>
<tr>
<td>Chatbot</td>
<td>Various methods</td>
<td>The Office</td>
</tr>
<tr>
<td>Analysis of Twitch chat</td>
<td>Clustering + classification</td>
<td>Twitch chat</td>
</tr>
<tr>
<td>Sentiment classification</td>
<td>Logistic/RNN/BER</td>
<td>Kaggle reviews</td>
</tr>
<tr>
<td>Detect fake news</td>
<td>RNN</td>
<td>COVID-19 articles</td>
</tr>
<tr>
<td>Clustering COVID-19 papers</td>
<td>Various clustering algs</td>
<td>CORD-19</td>
</tr>
<tr>
<td>Music recommendation</td>
<td>Various methods</td>
<td>MillionSong+Spotify</td>
</tr>
<tr>
<td>Text generation</td>
<td>RNNs, word2vec, + others</td>
<td>TrumpTweet dataset</td>
</tr>
<tr>
<td>Translation</td>
<td>RNNs and transformers</td>
<td>Japanese-English corpus</td>
</tr>
<tr>
<td>Lyrics generation</td>
<td>Transformers</td>
<td>Song Lyrics/Kaggle</td>
</tr>
<tr>
<td>Detect fake news</td>
<td>BOW classifiers</td>
<td>AAAI-21 COVID-19</td>
</tr>
<tr>
<td>Movie review predictions</td>
<td>Various classifiers + regression</td>
<td>IMDB+RottenTomatoes</td>
</tr>
<tr>
<td>Image to LaTeX code</td>
<td>Deep networks</td>
<td>im2latex dataset</td>
</tr>
<tr>
<td>Study of bias in embeddings</td>
<td>Bias detection/mitigation methods</td>
<td>WinoBias</td>
</tr>
</tbody>
</table>
Other examples of Projects from past CS 175 Classes

• Automatically matching resumes to jobs

• Predicting what subreddit a Reddit post belongs to or its future number of upvotes

• Identifying the genre of a movie or song

• Rating the quality of answers to questions on StackOverflow

• Automatically detecting emotion on Twitter or social media
  - E.g., tracking emotion in the context of certain hashtags over time and/or geographically

• Detecting fake product reviews

• Many many others.....
Questions?
Language Modeling Project: Comparing N-grams and RNNs

• **Task:** predict the most likely next word in text
  – Reading: Chapters 3 and 7 in Jurafsky and Martin

• **Technical approaches**
  – Baseline: Ngrams, Advanced: RNNs/Transformers

• **Datasets:**
  – Any large corpus, e.g., all or part of Wikipedia

• **Evaluation**
  – Rank words by probability of being next: higher rank for actual word is better
  – Log probability of next word given preceding words (or perplexity)

• **Enhancements**
  – Doing transfer learning on a domain-specific corpus
Question-Answering System

- **Task**
  - Generate an answer from within a document to specific types of questions
  - Reading: Chapter 23 in Jurafsky and Martin

- **Technical approaches**
  - RNNs, Encoder-decoder architectures

- **Datasets**
  - Multiple well-known Q/A data sets
    - e.g., see https://paperswithcode.com/task/question-answering

- **Evaluation**
  - Accuracy relative to known answers (exact match, close match, etc)

- **Extensions**
  - Can models “know what they don’t know”?
  - e.g., see this research paper: https://aclanthology.org/2021.findings-emnlp.385/
Automated Summarization of a Set of Documents

• Task
  – Automatically summarize documents in a corpus, e.g., reviews of restaurants
  – Reading, e.g.,
    • See others under Text Summarization on Course Website [www.ics.uci.edu/~smyth/courses/cs175/project_reading.html](http://www.ics.uci.edu/~smyth/courses/cs175/project_reading.html)

• Technical approaches
  – Various summarization algorithms

• Datasets
  – Reviews, legal documents, news articles, scientific papers: any large set of documents with common topics

• Evaluation
  – ROUGE method; also human user studies

• Extensions
  – Combine translation and summarization, e.g., summarize Chinese docs in English
Analysis of Aspects of Positive/Negative Reviews

• **Task**
  – Using labeled review data:
    • learn a classifier to predict positive/negative sentiment from the review (easy)
    • Combine with summarization: automatically extract what aspects of a product/movie/restaurant review are the basis for the positive or negative reviews (much harder to do well)
  – **Reading**: parts of chapters 5 to 9, 11, 17, 20 in Jurafsky and Martin

• **Technical approaches**
  – Part-of-speech parsing, Information extraction, Sentiment classification (logistic, RNNs, etc)

• **Datasets**: Yelp reviews, Amazon reviews, Movie reviews, etc

• **Evaluation**
  – Accuracy for classification part; ROUGE or human user study for aspects part

• **Extensions**
  – Compare lexicon-based methods with machine learning approaches
  – Build model on one corpus (e.g., Yelp) and see how well it works on another (e.g., IMDB)
Chatbot Project

• Task
  – Automatically generate the next (response) utterance in a conversation
  – Reading: Chapter 7, 9, 11, 24 in Jurafsky and Martin

• Technical approaches
  – Information retrieval methods
  – Neural architectures (encoder-decoder, transformers)

• Datasets
  – Switchboard, movie scripts, + several others

• Evaluation, e.g., system A versus B
  – Human user study (a type of Turing test: which responses are more real?)

• Enhancements
  – Build a domain-specific chatbot
  – Investigate how well a chatbot can track long-term context in a conversation
Chatbots

• Chatbot Setup
  – Human and agent alternate utterances (written or spoken)
  – Goal of agent is to generate an appropriate response utterance at each turn
    • conditioned on the history of the dialog up to that point

• Variations
  – Single-turn
    • Respond only to the last utterance
    • Will not be able to maintain coherence well (i.e., will “forget” earlier information)

  – Multiturn
    • Takes as input the previous K utterances (usually small for computational reasons)
Approaches to Chatbot Systems

• Rule-based
  – Early work in 1960’s, still widely-used today
  – Based on manually-defined pattern-matching + rules and slot-filling

• Corpus-based
  – Builds models for large corpora. Two general types
    • **Retrieval-based**: finds best-matching response utterance from a corpus
    • **Generative models**: generate a new response utterance
      – generator learned via machine learning, e.g., via an encoder-decoder RNN

• Real-world chatbots often use a combination of techniques, e.g., Alexa
Information Retrieval Response Generation

- Given a large corpus C with utterances t, and given a user’s sentence q
  - Define similarity sim(t, q), for all corpus utterances t in C
  - e.g., cosine similarity over bag-of-words/tf-idf or over sentence embeddings
  - E.g., maximize sim(t, q) and return utterance/response that came after t

- Many possible extensions....
  - Let q be the whole dialog so far
  - Query reformulation for q (for questions)

- Corpus C can be dialog turns or even non-dialog sentences
  - Transcripts of actual conversations, movie scripts
  - Wikipeida articles, news articles
Generative Responses, e.g., with Encoder-Decoders

From Jurafsky and Martin, 3rd edition, 2020
Generative Responses, e.g., with Encoder-Decoders

- **Issues with basic encoder-decoder architectures**
  - Tendency to produce predictable/short responses, e.g., “ok” or “I don’t know”
  - Inability to model longer context of conversation
  - Can lack coherence in responses across multiple turns
  - See Jurafsky and Martin for suggestions on how to handle these issues
  - Better performance with more recent neural models, e.g., Transformers

From Jurafsky and Martin, 3rd edition, 2020
Enhancements (applicable to multiple projects)

• Compare text representations
  – Different types of word embeddings (BERT, word2vec) to bag of words

• Compare character embeddings to word embeddings

• Systematically investigate effect characteristics of document on performance
  – e.g, performance metric = accuracy or precision
  – characteristic of document = number of words, fraction of different parts of speech, grammatical structure, etc

• Add a speech recognition front-end

• Evaluate domain transfer
  – Train a model on one domain, evaluate on a different domain, perhaps with some transfer learning
    • How well does a sentiment classifier for restaurant reviews “transfer” to tweets?
Using More Advanced NLP Techniques

• Question: What if you would like to propose using a new technique that you have no past experience with, e.g., Transformer models?
  – How can you write a proposal about a technique you are just starting to learn?

• Answer
  – Get at least a basic understanding of the technique before you write your proposal, e.g., read the relevant chapter in Jurafsky and Martin and/or one of the other references in the slides
  – In your proposal convey that you understand some of the basic concepts
  – It’s fine not to fully understand (yet) the details of a new technique
    …a major goal of a project class is that you will “learn by doing”
Questions?
Project Proposals
Project Proposals

• Your proposal should be 2 to 3 pages long
  – Required to use project proposal template (on the course Website)
  – One student submits the proposal, all students get the same grade as the team

• Project proposals will be graded like a homework assignment and receive a weight of 20% of your overall grade.

• Proposals will primarily be graded on
  – (a) clarity (is it clear what will be done in this project?) and
  – (b) completeness (does the proposal address all of the important aspects of the proposed project?)

• Note: if a project is too simple (or too complex), or missing important details, it may be returned to you and a revision requested.
Contents of Project Proposal

1. Project Summary
2. Proposed Technical Approach
3. Data Sets
4. Experiments and Evaluation
5. Software
6. Milestones
7. Individual Student Responsibilities
1. Project Summary

A clear description (1 to 2 sentences) that summarizes your project: e.g., “This project will use XX methods to predict YY using the Z1 and Z2 data sets, with evaluation using classification accuracy and user studies.”

Examples:

- The goal of this project is to explore spam filtering by classifying SMS messages as spam or not spam using various machine learning techniques.

- Application and evaluation of multiple sentiment analysis classification algorithms using multiple datasets. Evaluation of correctness, uncertainty, and differences between the algorithms.

- Our project will be a poem generator that takes in a dictionary and outputs a poem that looks as if a human created the poem. We will be using various algorithms and libraries, such as NLTK and NodeBox, to identify different grammars and apply constraint satisfaction to make a poem that is clear and readable.
2. Proposed Technical Approach

- Write a paragraph with a clear description of the methods and algorithms you plan to use on the project.

- If the system you are building can be thought of as a pipeline with multiple components feel free to provide a figure that illustrates the pipeline with blocks for different components and brief descriptions of each component.

- Make sure you are clear on what the inputs and outputs are for your system:
  - Overall end-to-end inputs and outputs
  - Inputs and outputs for individual components
  - This will help you in thinking about your system and coordinating your team.
2. Proposed Technical Approach (Example)

We are going to divide the whole project into several stages.

- The first stage is to preprocess the movie review. We plan to use stopwords list from NLTK to remove stopwords, punctuations and non-alphabetic words. Secondly, we plan to obtain the base part of the word by using Snowball as a stemmer to remove morphological endings. The Porter Stemming Algorithm will be implemented in this step. Then, we plan to extract opinioned words by applying the positive words list from NLTK.

- The second stage is to classify the opinioned words list obtained from the first stage.

- The third stage is training our classifier to recognize the attitude of the reviews. We plan to try logistic regression, support-vector machines, and Multinominal Naive Bayes models for supervised training.
2. Proposed Technical Approach (Example)

Our Algorithm

For each unique food product ID, we look at every sentence of individual reviews and remove all the stopwords (Python NLTK stopwords and our custom set of stopwords). This stopword removal step ensures that irrelevant, unhelpful key phrases do not get added to our set of key phrases. We then extract all n-grams that begin with an adjective and end with a noun (and vice versa) from the filtered reviews. We rank these key phrases based on their values of PF-IRF (phrase frequency - inverse review frequency), which is a variation of TF-IDF (term frequency - inverse document frequency). Finally, we can generate new sentences using only the key phrases with high rankings, along with a Markov model simulating common sentence structures.
2. Proposed Technical Approach (Example)
2. Proposed Technical Approach (Example)
2. Proposed Technical Approach (Example)
Questions?
3. Data Sets

- Briefly describe what data set(s) you plan to use in the project. Include references to the data (e.g., a URL) if you can. ......

- If you are able to access and take an initial look at your data, feel free to also include a figure or two in this section, e.g., a histogram of document lengths.

- You can change or add to your data sets during the project if you wish to, but you should have identified at least one data set to work with by the time you submit the proposal.
3. Data Sets (Example)

- We plan to work with the Rotten Tomatoes movie review dataset publicly provided on Kaggle.com as the basis for a machine learning competition. The data set is a collection of sentences from reviews that are parsed into phrases by the Stanford Parser. The data set is preprocessed with a predefined vocabulary that simply removes repeated common or short phrases. We plan to improve this simple vocabulary through our own method of preprocessing as discussed in the next section.
3. Data Sets (Example for a Chatbot Project)

Cornell Movie--Dialogs Corpus\(^1\) (phase two)
When constructing the chatbot, simple data sets from movie dialogs would be used. In this dataset, 220,579 conversational exchanges between 10,292 pairs of movie characters were recorded.

Amazon question/answer data\(^2\) (phase three)
This dataset was extracted from Question-and-Answer sections. There are around 1.4 million answered questions for all categories of products. In addition, some portions of the datasets would contain questions with multiple answers.

ConvAI2 Competition Dataset\(^3\) (phase three)
This dataset was used in the ConvAI2 Competition in 2020. In this dataset, there are a total 3,929 questions and 11,489 single-turn conversations. In the training dataset, topics, facets, questions, answers, and clarification need labels were included.

Topical-Chat\(^4\) (phase three)
This dataset contains conversation files and reading set files with 10,784 conversations and 235,434 utterances. Each conservation would have a conversation rating where the Turker rates the quality of the conversation.
## 3. Data Sets (Example)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Length</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bible Corpus [4]</td>
<td>Dataset containing ASV, BBE, DARBY, KJV, WBT, WEB, YLT versions of the bible</td>
<td>Roughly 783,137 words per version</td>
<td>Religious text (verse indexable)</td>
</tr>
<tr>
<td>Helsinki Corpus of English Texts (HC) [5]</td>
<td>Corpus of English texts from 730 - 1710 in xml format</td>
<td>1,572,800 words 450 texts</td>
<td>Poetry/Prose, Religious Texts, Epic Poetry, etc.</td>
</tr>
<tr>
<td>Corpus of Middle English Prose and Verse [6]</td>
<td>Corpus of Middle English primary texts Need-to-scrape</td>
<td>300 Middle English primary texts</td>
<td>Poetry/Prose, Religious Texts, Epic Poetry, etc.</td>
</tr>
</tbody>
</table>
### 3. Data Sets (Example)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Length</th>
<th>Type</th>
<th>Classified</th>
</tr>
</thead>
</table>
| Yelp RSS feed         | http://www.yelp.com/rss  
25 reviews for 5 cities per day (125)  
Need to collect (1000+) | Short: around 300 characters   | Review       | Yes: each has a rating (N/5) |
| Song Lyrics           | http://www.songlyrics.com/  
Need to collect | Medium: Around 2,500 characters | Song, Poetry | No                           |
| Twitter Tweets        | http://www.sananalytics.com/lab/twitter-sentiment/  
5513 tweets  
Pre-packaged | Short: max 140 characters | Message      | Yes: hand classified         |
| Movie Reviews         | http://www.cs.cornell.edu/people/pabo/movie-review-data/  
polarity dataset v2.0  
1000 positive, 1000 negative reviews  
Pre-Packaged | Medium-Long                  | Review       | Yes: positive/negative      |
3. Data Sets (Example)

![Bar chart showing Top 10 Subreddits by Comment in May 2015](image-url)
3. Data Sets (Example)
Questions?
4. Experiments and Evaluation

- Provide a brief and clear description of how you will evaluate the results of your project, e.g., accuracy for classification, precision-recall for document ranking.

- Aspects to consider
  - Single metrics: classification accuracy
  - Curves: precision-recall
  - Test sets and cross-validation
  - Manual analysis of the types of error a system makes
  - User studies

(in next week’s Wednesday lecture we will discuss evaluation in more detail)
4. Experiments and Evaluation (Examples)

- The data set comes in a set of two files: one train.tsv file and one test.tsv file. We will essentially use cross validation to split our training data set into a validation set and a testing set in order to evaluate our models. We could also use our models on the test.tsv file and upload a submission file onto Kaggle, which will evaluate our models and give us a corresponding score on the leaderboard. The Kaggle leaderboard currently has about 700 teams, who are ranked by how well their model performs on the test data provided in test.tsv - this can give us an accurate indication on how well our model performs in a more realistic setting.
5. Software

• Provide a list of the major pieces of project software that you expect to use, divided into 2 sets:
  – (1) publicly-available code, and
  – (2) code will write yourself.

• This list will probably be incomplete at this point (which is fine) since you may not know yet about all of the publicly-available software that might be relevant to your project
5. Software (Example)

Publicly-available code:

- **NLTK**: provides a list of stop words and built-in naive Bayes classifier.
- **PyEnchant**: provides spell checking and spelling suggestions.
- More later.

Code to be written ourselves in Python:

- **Tokenizer** to parse SMS message.
- **Generate a feature list using tokenized message**.
- **Group misspelled words using PyEnchant suggestions**.
- **Track metadata such as number of misspellings in a message**.
- **Bernoulli naive Bayes** to analyze features and classify messages.
## 5. Software (Example)

<table>
<thead>
<tr>
<th>Publicly-Available Code</th>
<th>Code We Will Write/Test</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Programming Languages:</strong></td>
<td><strong>Ranking Algorithms:</strong></td>
</tr>
<tr>
<td>• Python 3.5 and libraries such as NLTK, NumPy, SciPy, and Matplotlib</td>
<td>• Determine the phrase rank for a product by computing the value of PF-IRF measure for each phrase. Phrases with high PF-IRF ranks are selected and included in our summary.</td>
</tr>
<tr>
<td>• SQLite3</td>
<td>• <strong>Comparison algorithm</strong> to evaluate similarity between RAKE’s phrases and our algorithm’s phrases.</td>
</tr>
<tr>
<td><strong>Keywords Extraction:</strong></td>
<td><strong>Phrase Extraction Algorithm:</strong></td>
</tr>
<tr>
<td>• RAKE (Rapid Automatic Keyword Extraction) using NLTK</td>
<td>• Extract frequent n-grams from all reviews of a product.</td>
</tr>
<tr>
<td><strong>Evaluation Software:</strong></td>
<td>• Extract key phrases from the frequent n-grams, phrases that contain adjectives followed by nouns (and vice versa) with stopwords eliminated.</td>
</tr>
<tr>
<td>• ROUGE software package to automate the evaluation of our results.</td>
<td><strong>Sentence Generation Algorithm:</strong></td>
</tr>
<tr>
<td></td>
<td>• Uses sentence structure and a <strong>Markov chain</strong> to generate readable sentences containing descriptive phrases.</td>
</tr>
</tbody>
</table>
6. Milestones

- Provide a brief list of milestones. For example, since the project will span 6/7 weeks of the class, you could break your milestones into a list of 3 intermediate phases:
  - Weeks 4, 5 and 6
  - Weeks 7 and 8
  - Weeks 9 and 10

- For example, much of the data gathering and preprocessing and coding (development and test) could happen in the earlier weeks, and much of the experimentation and evaluation in the later weeks. Note that you have a progress report due at the end of week 7.
6. Milestones (Example)

Weeks 4 and 5:
- *Search for additional data sets*
- *Write basic tokenizer and generate feature list using most common words.*
- *Test accuracy of scikit-learn Bernoulli naive Bayes and multinomial naive Bayes using the basic tokenizer.* *(If multinomial turns out to be more accurate, then we will implement multinomial instead).*

Weeks 6, 7 and 8:
- *Write Bernoulli naive Bayes classifier, and test accuracy of basic algorithm*
- *Compare accuracy with scikit-learn Bernoulli naive Bayes and Multinomial naive Bayes.*
- *Improve tokenizer and feature list generation.*
- *Use PyEnchant to group misspelled words.*

Weeks 9 and 10:
- *Improve tokenizer and feature list generation.*
- *Explore tracking different metadata features.*
7. Individual Student Responsibilities

Summarize briefly what each student will be primarily responsible for in the project. For example, you might write something like this:

• Name 1: will define an overall system architecture and pipeline, will be responsible for integration of components into the pipeline, will do research and acquire data sets to train and evaluate the system, will be responsible for preprocessing data (e.g., define the vocabulary for the algorithms), will assist in doing experiments and interpreting results, will assist in writing project reports

• Name 2: will conduct research on technical approaches, will develop and test the code for team’s new Algorithm, will integrate new Algorithm into the pipeline, will assist in doing experiments and interpreting results, will assist in writing project reports

• Name 3: will implement a baseline approach and integrate it into the pipeline, will conduct research on evaluation methods, will be responsible for overall evaluation of the algorithms and system, will coordinate and take the lead in writing project reports.

[Note these are just suggestions – you can and should organize responsibilities in whatever way makes sense.....]
Example of References (Optional)

9. References & Links


Questions?
Tips on Planning and Organizing Projects
Project Tips: Goals

• Be clear in your high-level goals
  – e.g., “will systematically evaluate the accuracy of logistic regression and neural network classifiers on the Reuters data set and 2 other data sets”
  – Ok to not to have all the details of how you will get there, but important to know what the goal is
  – Ok if goals are updated/changed as you learn more about the problem

• Team members should agree on the goals
  – To be effective the team needs to be clear about the goals
  – If there is any doubt about what the goal is, ask questions and discuss
Project Tips: Plan in Stages

Plan your project in stages so that the overall project is not dependent on the riskier elements working

Example:

PHASE 1

<table>
<thead>
<tr>
<th>Original Documents</th>
<th>Standard Bag of Words</th>
<th>Standard Logistic Regression</th>
<th>Cross-Validation Experiments</th>
</tr>
</thead>
</table>
Project Tips: Plan in Stages

Plan your project in stages so that the overall project is not dependent on the riskier elements working

Example:

PHASE 1
Original Documents → Standard Bag of Words → Standard Logistic Regression → Cross-Validation Experiments

PHASE 2
Pretrained Embeddings → Standard Logistic Regression
Project Tips: Plan in Stages

Plan your project in stages so that the overall project is not overly dependent on the riskier elements

Example:

PHASE 1
Original Documents → Standard Bag of Words → Standard Logistic Regression → Cross-Validation Experiments

PHASE 2
Pretrained Embeddings

PHASE 3
Feedforward Network
Project Tips: Plan in Stages

Plan your project in stages so that the overall project is not overly dependent on the riskier elements

Example:

PHASE 1
Original Documents

PHASE 2
Pretrained Embeddings

PHASE 3
Standard Bag of Words

PHASE 4
Standard Logistic Regression

Cross-Validation Experiments

Fine-tuned Transformers

Feedforward Network
Project Tips: Evaluation Methods

- Very important to have a clear idea of how you will evaluate your system

- For some tasks, such as document classification, there are well-defined metrics that are straightforward
  - E.g., cross-validated classification accuracy

- For other tasks, such as clustering, you will have to do some research to figure out what metrics are appropriate
  - For some projects, some user evaluation may be necessary

- Always include a baseline method in your experiments
  - E.g., for classification your baseline could be a Naïve Bayes classifier
Project Tips: Revision/Source Control for Code

- **Teams should consider using a collaborative revision control system**
  - e.g., Github accounts (freely available)
  - If you are not familiar with these tools, this is a good opportunity to learn

- **Revision control**
  - Provides a systematic way for a team to develop code, scripts, documents, etc
  - Individuals can “check out” code, work on it, and then “commit”
  - Earlier versions of code can be recovered
    - Useful when you want to go back to an earlier version without a bug
Project Tips: Revision Control for Experiments

• You are likely to conduct many experiments over the course of the project, comparing versions of preprocessing, parameter settings, algorithms
  – You want to be able to keep track of your experiments and results in a systematic way

• Recommendations:
  – Organize and document your code/scripts for experiments
  – Use time-stamps, give your scripts interpretable names, use comments

• Consider using tools such as Jupyter Notebooks, with github, for documentation and collaboration
  – just like a “lab notebook” in a science lab

• Reproducibility is important
  – Scripts will allow to you to regenerate results from earlier in the project
  – Also allows team members to share results efficiently
General Project Tips: Writing

• Write clearly
  – Try to put yourself in the mind of the person reading it
  – Make sure you don’t leave out important details and concepts
  – Use spell-checkers, grammar-checkers, etc
  – Target audience = a senior in CS or someone with a CS degree
  – Main issue where students lose points in grading? Description/writing is not clear

• Work collaboratively
  – Teams can use a shared document with version control, e.g., Google Docs
  – Have each member of the team edit, read, comment on the document
  – Work on your documents collaboratively: will lead to better documents

• Iterate, iterate, iterate: good writing comes from revising

• Use figures and tables when you can
  – “A picture is worth a 1000 words”

• Don’t cut and paste text from other courses without citing it
Wrapup
Announcements

- **Office hours:**
  - Instructor: Thursdays, 4 to 5:30
    - See online Google signup sheet (link available in Ed)
  - Sakshi (TA): Fridays 10 to 11 and Mondays 10 to 11

- **Discussion sections** with TA Sakshi: Thursday 1 to 2, 2 to 3
  - Students can discuss project interests, teaming up: moderated by TA

- **EdD Discussion Board**
  - Post questions about possible projects, questions from lectures, etc

- **Project Proposal**
  - Due Tuesday night next week
Project Proposals

- Due Tuesday night next week, worth 20% of your grade

- Find a project team
  - Use Google Sheet, reach out to team members, go to discussion section Thursday

- Decide on a project topic with your team

- Review instructions for proposals

- Write your proposal
  (we will review the structure of proposals later in this lecture)
Finding Project Teams

- Make sure your information is in the Google Sheet

- If you are a 3-person team already:
  - Make sure everyone’s name is associated with the team

- If you are 2 students looking for a 3rd
  - Find individuals with similar interests in the Google Sheet and email them

- If you are 1 student looking for 2 more
  - Find people with similar interests in the Google Sheet and email them
  - It’s fine to find just 1 person initially, and then search for a 3rd
  - Feel free to contact any 2-person teams in the Sheet

- Both individuals and 2-student groups
  - Attend Discussion Section on Thursday to find candidate team members

IMPORTANT: please update the sheet every time your individual or team status changes. This is important so that we can track who is on a team and who is not.
Lectures Next Week

• **Monday**
  – Review of project proposals, answer questions
  – Additional topics
    • Word embeddings
    • Large language models such as Transformers
    • Fine-tuning large language models
    • Software libraries such as HuggingFace

• **Wednesdays**
  – More on neural language models and applications
  – Evaluation methods (useful for projects)
Questions? (with video recording off)