School of Computing and Information Systems
The University of Melbourne
COMP90042 Natural Language Processing (Semester 1, 2024)

Sample solutions: Week 3

Discussion

1. What is **text classification**? Give some examples.

   - Numerous examples from the lectures: sentiment analysis, author identification, automatic fact-checking, etc.

   (a) Why is text classification generally a difficult problem? What are some hurdles that need to be overcome?

   - The main issue is in terms of **document representation** — how do we identify **features** of the document which help us to distinguish between the various classes?
   - The principal source of features is based upon the presence of tokens (words) in the document (known as a **bag–of–words** model). However, many words don't tell you anything about the classes we want to predict, hence **feature selection** is often important. On the other hand, single words are often inadequate at modelling the meaningful information in the document, but multi-word features (e.g. bi-grams, tri-grams) suffer from a **sparse data problem**.

   (b) Consider some (supervised) text classification problem, and discuss whether the following (supervised) machine learning models would be suitable:

   - The answers will vary depending on the nature of the problem, the feature set, the class set, and so on. One possible solution, for a generic genre identification problem using an entire bag–of–words model (similar to the notebook) is as follows:

   i. k-Nearest Neighbour using Euclidean distance

   - Often this is a bad idea, because Euclidean distance tends to classify documents based upon their **length** — which is usually not a distinguishing characteristic for classification problems.

   ii. k-Nearest Neighbour using Cosine similarity

   - Usually better than the previous, because we're looking at the distribution of terms. However, k-NN suffers from high-dimensionality problems, which means that our feature set based upon the presence of (all) words usually isn't suitable for this model.

   iii. Decision Trees using Information Gain

   - Decision Trees can be useful for finding meaningful features, however, the feature set is very large, and we might find **spurious** correlations. More fundamentally, Information Gain is a poor choice because it tends to prefer **rare** features; in this case, this would correspond to features that appear only in a handful of documents.

   iv. Naive Bayes

    - At first glance, a poor choice because the assumption of the **conditional independence** of features and classes is highly untrue.
    - Also sensitive to a large feature set, in that we are multiplying together many (small) probabilities, which leads to biased interpretations based upon otherwise uninformative features.
    - Surprisingly somewhat useful anyway!

  v. Logistic Regression
    - Useful, because it relaxes the conditional independence requirement of Naive Bayes.
    - Since it has an implicit feature weighting step, can handle large numbers of mostly useless features, as we have in this problem.

  vi. Support Vector Machines
    - Linear kernels often quite effective at modelling some combination of features that are useful (together) for characterising the classes.
    - Need substantial re-framing for problems with multiple classes (instead designed for two-class (binary) problems); most text classification tends to be multi–class.

2. For the following "corpus" of two documents:

```
1.  how much wood would a wood chuck chuck if a wood chuck
would chuck wood
2.  a wood chuck would chuck the wood he could chuck if
a wood chuck would chuck wood
```

- I'm going to show the frequencies of the 11 different word uni-grams below, as it will make life a little easier in a moment:

a	chuck	could	he	how	if	much	the	wood	would	</s>	Total
4	9	1	1	1	2	1	1	8	4	2	34

(a) Which of the following sentences: `A`: `a wood could chuck`; `B`: `wood would a chuck`; is more probable, accoding to:

  i. An unsmoothed uni-gram language model?
    - An unsmoothed uni-gram language model is simply based on the counts of words in the corpus. For example, out of the 34 tokens (including `</s>`) in the corpus, there were 4 instances of `a`, so $P(\texttt{a}) = \frac{4}{34}$
    - Note that technically we can choose to *exclude* `</s>` here, as it would not have any impact when it comes to ranking sentences (e.g. a unigram model would score `a cat meows` and `meows cat a` the same, whether we include `</s>` or not even though the former sentence is a more fluent sentence that terminates correctly). Here we choose to include `</s>` for consistency (as it will be used for higher order n-gram models in subsequent questions).

- To find the probability of a sentence using this model, we simply multiply the probabilities of the individual tokens:

$$\begin{aligned} P(A) &= P(\texttt{a})P(\texttt{wood})P(\texttt{could})P(\texttt{chuck})P(\texttt{</s>}) \\ &= \frac{4}{34} \times \frac{8}{34} \times \frac{1}{34} \times \frac{9}{34} \times \frac{2}{34} \approx 1.27 \times 10^{-5} \\ P(B) &= P(\texttt{wood})P(\texttt{would})P(\texttt{a})P(\texttt{chuck})P(\texttt{</s>}) \\ &= \frac{8}{34} \times \frac{4}{34} \times \frac{4}{34} \times \frac{9}{34} \times \frac{2}{34} \approx 5.07 \times 10^{-5} \end{aligned}$$

- Clearly sentence B has the greater likelihood, according to this model.

ii. A uni-gram language model, with Laplacian ("add-one") smoothing?

- Recall that in add-one smoothing, for each probability, we add 1 to the numerator and the size of the vocabulary, which is 11, to the denominator. For example, $P_{\mathrm{L}}(\texttt{a}) = \frac{4+1}{34+11} = \frac{5}{45}$.
- Everything else proceeds the same way:

$$\begin{aligned} P_{\mathrm{L}}(A) &= P_{\mathrm{L}}(\texttt{a})P_{\mathrm{L}}(\texttt{wood})P_{\mathrm{L}}(\texttt{could})P_{\mathrm{L}}(\texttt{chuck})P_{\mathrm{L}}(\texttt{</s>}) \\ &= \frac{5}{45} \times \frac{9}{45} \times \frac{2}{45} \times \frac{10}{45} \times \frac{3}{45} \approx 1.46 \times 10^{-5} \\ P_{\mathrm{L}}(B) &= P_{\mathrm{L}}(\texttt{wood})P_{\mathrm{L}}(\texttt{would})P_{\mathrm{L}}(\texttt{a})P_{\mathrm{L}}(\texttt{chuck})P_{\mathrm{L}}(\texttt{</s>}) \\ &= \frac{9}{45} \times \frac{5}{45} \times \frac{5}{45} \times \frac{10}{45} \times \frac{3}{45} \approx 3.66 \times 10^{-5} \end{aligned}$$

- Notice that the probability of sentence A is larger using this model, because the probabilities of the unlikely `could` and `</s>` have increased. (The other probabilities have decreased). Sentence B is still more likely, however.

iii. An unsmoothed bi-gram language model?

- This time, we're interested in the counts of pairs of word tokens. For example, the probability of the bi-gram `wood would` is based on the count of that sequence of tokens, divided by the count of `wood`: $\frac{1}{8}$ (because only a single `wood` is followed by `would`).
- We include sentence terminals, so that the first probability in sentence A is $P(\texttt{a}|\texttt{<s>}) = \frac{1}{2}$ — because one of the two sentences in the corpus starts with `a`. We also need to predict $P(\texttt{</s>}|\texttt{chuck}) = \frac{0}{9}$ — because none of the 9 `chuck`s are followed by the end of the sentence.
- Now, we can substitute:

$$\begin{aligned} P(A) &= P(\texttt{a}|\texttt{<s>})P(\texttt{wood}|\texttt{a})P(\texttt{could}|\texttt{wood})P(\texttt{chuck}|\texttt{could})P(\texttt{</s>}|\texttt{chuck}) \\ &= \frac{1}{2} \times \frac{4}{4} \times \frac{0}{8} \times \frac{1}{1} \times \frac{0}{9} = 0 \\ P(B) &= P(\texttt{wood}|\texttt{<s>})P(\texttt{would}|\texttt{wood})P(\texttt{a}|\texttt{would})P(\texttt{chuck}|\texttt{a})P(\texttt{</s>}|\texttt{chuck}) \\ &= \frac{0}{2} \times \frac{1}{8} \times \frac{1}{4} \times \frac{0}{4} \times \frac{0}{9} = 0 \end{aligned}$$

- Because there is a zero–probability element in both of these calculations, they can't be nicely compared, leading us to instead consider:

iv. A bi-gram language model, with Laplacian smoothing?

- We do the same idea as uni-gram add–one smoothing. The vocabulary size is 11.

$$\begin{aligned}P_{\text{L}}(A) =& P_{\text{L}}(\texttt{a}|\texttt{<s>})P_{\text{L}}(\texttt{wood}|\texttt{a})P_{\text{L}}(\texttt{could}|\texttt{wood})P_{\text{L}}(\texttt{chuck}|\texttt{could})\\ & P_{\text{L}}(\texttt{</s>}|\texttt{chuck})\\ =&\frac{2}{13}\times\frac{5}{15}\times\frac{1}{19}\times\frac{2}{12}\times\frac{1}{20}\approx 2.25\times10^{-5}\\ P_{\text{L}}(B) =& P_{\text{L}}(\texttt{wood}|\texttt{<s>})P_{\text{L}}(\texttt{would}|\texttt{wood})P_{\text{L}}(\texttt{a}|\texttt{would})P_{\text{L}}(\texttt{chuck}|\texttt{a})\\ & P_{\text{L}}(\texttt{</s>}|\texttt{chuck})\\ =&\frac{1}{13}\times\frac{2}{19}\times\frac{2}{15}\times\frac{1}{15}\times\frac{1}{20}\approx 3.60\times10^{-6}\end{aligned}$$

- This time, sentence A has the greater likelihood, mostly because of the common bi-gram `a wood`.

v. An unsmoothed tri-gram language model?

- Same idea, longer contexts. Note that we now need two sentence terminals.

$$\begin{aligned}P(A) &= P(\texttt{a}|\texttt{<s> <s>})P(\texttt{wood}|\texttt{<s> a})\cdots P(\texttt{</s>}|\texttt{could chuck})\\ &= \frac{1}{2}\times\frac{1}{1}\times\frac{0}{4}\times\frac{0}{0}\times\frac{0}{1}=?\\ P(B) &= P(\texttt{wood}|\texttt{<s> <s>})P(\texttt{would}|\texttt{<s> wood})\cdots P(\texttt{</s>}|\texttt{a chuck})\\ &= \frac{0}{2}\times\frac{0}{0}\times\frac{1}{1}\times\frac{0}{1}\times\frac{0}{0}=?\end{aligned}$$

- Given that the unsmoothed bi-gram probabilities were zero, that also means that the unsmoothed tri-gram probabilities will be zero. (Exercise for the reader: why?)
- In this case, they aren't even well–defined, because of the $\frac{0}{0}$ terms, but we wouldn't be able to meaningfully compare these numbers in any case.

vi. A tri-gram language model, with Laplacian smoothing?

- The vocabulary size is 11. Everything proceeds the same way:

$$\begin{aligned}P_{\text{L}}(A) &= P_{\text{L}}(\texttt{a}|\texttt{<s> <s>})P_{\text{L}}(\texttt{wood}|\texttt{<s> a})\cdots P_{\text{L}}(\texttt{</s>}|\texttt{could chuck})\\ &= \frac{2}{13}\times\frac{2}{12}\times\frac{1}{15}\times\frac{1}{11}\times\frac{1}{12}\approx 1.30\times10^{-5}\\ P_{\text{L}}(B) &= P_{\text{L}}(\texttt{wood}|\texttt{<s> <s>})P_{\text{L}}(\texttt{would}|\texttt{<s> wood})\cdots P_{\text{L}}(\texttt{</s>}|\texttt{a chuck})\\ &= \frac{1}{13}\times\frac{1}{11}\times\frac{2}{12}\times\frac{1}{12}\times\frac{1}{11}\approx 8.83\times10^{-6}\end{aligned}$$

- Notice that the problem of unseen contexts is now solved (they are just $\frac{1}{11}$).
- Sentence A has a slightly greater likelihood here, mostly because of the `a` at the start of one of the sentences (note that this will continue to be "seen" even at higher orders of n). You can also see that the numbers are getting very small, which is a good motivation for summing log probabilities (assuming no zeroes) rather than multiplying.

(b) Assuming we are using a bi-gram language model with Kneser-Ney smoothing. Given the bigram `chuck a`, compute the continuation probability for `a`.

- Observed bi-grams: `chuck chuck`, `chuck if`, `chuck would`, `chuck wood`, `chuck the`
- Unobserved bi-grams: `chuck a`, `chuck could`, `chuck he`, `chuck how`, `chuck much`, `chuck </s>`
- Continuation counts for word types that do no co-occur with `chuck`:
  - $\texttt{a} = \{\texttt{would}, \texttt{if}, \texttt{<s>}\} = 3$
  - $\texttt{could} = \{\texttt{he}\} = 1$
  - $\texttt{he} = \{\texttt{wood}\} = 1$
  - $\texttt{how} = \{\texttt{<s>}\} = 1$
  - $\texttt{much} = \{\texttt{how}\} = 1$
  - $\texttt{</s>} = \{\texttt{wood}\} = 1$
-

$$P_{cont}(\texttt{a}) = \frac{\#_{cont}(\texttt{a})}{\#_{cont}(\texttt{a}) + \#_{cont}(\texttt{could}) + \#_{cont}(\texttt{he}) + \#_{cont}(\texttt{how}) + \#_{cont}(\texttt{much}) + \#_{cont}(\texttt{</s>})}$$
$$= \frac{3}{3+1+1+1+1+1}$$

3. What does **back–off** mean, in the context of smoothing a language model? What does **interpolation** refer to?

- Back–off is a different smoothing strategy, where we incorporate lower–order n-gram models (in particular, for unseen contexts). For example, if we have never seen some tri-gram from our sentence, we can instead consider the bi-gram probability (at some penalty, to maintain the probability of all of the events, given some context, summing to 1). If we haven't seen the bi-gram, we consider the uni-gram probability. If we've never seen the uni-gram (this token doesn't appear in the corpus at all), then we need a so–called "0-gram" probability, which is a default for unseen tokens.
- Interpolation is a similar idea, but instead of only "falling back" to lower–order n-gram models for unseen events, we can instead consider every probability as a linear combination of all of the relevant n-gram models, where the weights are once more chosen to ensure that the probabilities of all events, given some context, sum to 1.