# SMU MSDS 7337 Mid-Term Exam Summer 2019

Korey MacVittie

# Short Essay Responses

(25 pts each, 300-500 words each).

1. Select one career or industry that makes use of applied NLP.
   1. Explain generally how that field or career utilizes NLP.
   2. Explain at least some methods of NLP that are very likely to be used in the career or industry you selected.
   3. Give at least one specific example of a use case for NLP within the chosen field, and explain how the problem or situation is (or could be) improved by applying NLP.

(316 words)

Machine translation is a field that makes significant use of NLP, by translating words or sentences in one language to another. While single word translation can be relatively straight-forward, sentence translation is significantly more complex. The machine must be able to understand a word's meaning, derived from context (for instance, the word "read" in English has multiple meanings, which must be derived from the sentence as a whole); it then must locate words that match or convey a similar thought in the target language, and reconstruct the sentence using the target language's grammar.

The primary means of doing this today involves tagging: identifying which part of speech a particular word is, to assist in both constructing the appropriate grammar in the target language, as well as determining which meaning of the word is to be used. This may also involve stemming: the process of finding the root word in a word (for instance, in “finding,” the root word is “find”), which is an important part of determining both the meaning and type of a word: rather than have many definitions for the word “find” and its variations, a dictionary could simply have “find” and list out its various possible endings, each indicating what type of speech it is, which would help retain the meaning of the word.

Translation as a whole benefits significantly from NLP. Even in its current state, we often find ourselves reliant upon individuals with the proper linguistic knowledge: for instance, I sometimes make use of English-French machine translation, but find myself making small adjustments to the result using my minimal French knowledge, and wouldn’t rely upon it for professional use. In a world in which it is far easier for conversations to happen between individuals who do not share a language, having tools that allow us to communicate readily will become more and more important, given the difficulty of learning new languages.

1. Choose one of the “trade-offs” in NLP that was covered in the materials for this course.
   1. Explain the trade-off in general terms. Define the two choices.
   2. Explain the benefits and weaknesses of each side of the trade-off. Include at least one benefit and one weakness of each.
   3. Describe a work-situation that would make one of the choices in the trade-off much better, in terms of practical outcomes for you and your stakeholders on a project.

(455 words)

One of the trade-offs in NLP is going either “deep” or “shallow.” The goal of shallow NLP is to retrieve pertinent information from the text, rather than diving into everything and parsing the entirety of the text. It may be possible that the sort of information being sought can be found without delving into the minutiae of the grammar, for instance, or perhaps the text is being skimmed only for particular words or phrases. In shallow NLP, the algorithm doesn’t have a strong “understanding” of the text, only specified pieces, parts, or levels of it.

Conversely, deep NLP seeks to tease out every detail and nuance of the text. At its peak, the goal of deep NLP is to understand every aspect of the text as well as, or better than, a human reader fluent in the relevant language. At present, this level of depth is beyond what we have achieved, as a species, with NLP; however, it should be understood that “shallow” and “deep” NLP aren’t binary, it is more of a continuum, with some applications existing today that seem to be approaching the deep end of the pool.

Shallow NLP, in general, has more direct utility in most use cases. An application does not need to understand every nuance of a text to tease out sentiment, for instance, nor does a precise mapping of the grammatical structure help if you are only looking for specific keywords (a bank’s phone system, for instance, most likely only needs a set of keywords to interact properly with calling customers; it doesn’t need to understand every word they utter).

Meanwhile, deep NLP – truly deep – has the current drawback of being not possible with our current understanding of technology and AI design. However, the advantages of such a system would be full, human-like comprehension of language: this is something akin to AI shown in science fiction, with mechanical beings capable of speech on the level of people. While there might not be much commercial use for such a thing (or at least, not any readily obvious to this writer), such an achievement would have monumental impact on humanity.

Deep NLP has specific use-cases where the additional information is helpful or needed: machine translation would be a good example where that level of detailed knowledge of the meanings and grammar involved in a sentence would be almost necessary to provide a good translation of the text. With only a shallow understanding of the text, machine translation provides awkward phrases, stilted or poor grammar, or outright mistranslations. Context clues are often important to narrowing down a given word’s meaning in a specific instance, and shallow NLP doesn’t provide the nuance or the power necessary to perform that sort of analysis.

# Essay Response

(50 pts, 600-1000 word essay).

1. Describe the overall NLP “pipeline” for a text analytics project as described in the materials of this course. First give an overview, then briefly describe and explain each phase of the pipeline.
2. For each phase of the pipeline, either (a) recommend a specific tool for getting the job done, e.g. a Python package, and a reason why you recommend it, or (b) explain a choice that must be made, where you would configure the process for that step very differently depending on the kind of application.

Note: For 2, do each of (a) and (b) at least once.

(880 words)

Not all NLP pipelines are equal; there are a number of different approaches that are valid, so I shall outline one method here.

The first step, of course, is data collection: without data, *data* science cannot happen. In the context of NLP, this implies some kind of corpus, a collection of text documents or similar language-based data on which the analysis is to be performed.

The next step would be sentence segmentation: breaking the data down into its component pieces. It is significantly easier to work with sentences, as opposed to paragraphs or even larger language “bits.” While this step is not wholly necessary. In some cases, may be undesired: if we are working with tweets from Twitter, for instance, it may be better to ignore this step, as a whole tweet could be considered a “chunk” of language, expressing a single idea, even if divided into sentences.

Once we have sentences, we then tokenize the words contained in them. “This is a sentence.” would be broken into ‘this,’ ‘is,’ ‘a’, ‘sentence’, and ‘.’, for example. We want to tokenize out punctuation, as those can be carriers of meaning; however, some punctuation, such as that used in contractions, we would want to consider as part of the word itself (“don’t”, for example, should not become ‘don’, ‘’’, ‘t’).

From here, we can begin tagging, or identifying the part of speech for each word. This step is somewhat tricky, as it can be reliant on context clues: for instance, “water” could be a noun (referring to the liquid), or it could be a verb (as in “to water one’s horse,” in which the word denotes the action of giving water to the animal). There are some awkward edge cases, such as gerunds, which can further complicate the process.

The next part of the process would be lemmatization, or stemming. This is the process of turning words into their roots, or changing to what might be called a base form: variations of “to be,” for instance, like “is,” could be transformed into their root, while a word like “horses” should be transformed into its singular, “horse.” This part of the process, I think, is somewhat up to debate: it could well be argued that “horses” and “horse,” while having the same linguistic root, have different meanings (there is a distinct conceptual difference between “multiple horses” and “singular horse”), but to my knowledge there isn’t a process that deals with this, and I would be uncertain how to handle it. In the case of “horses,” one could perhaps replace it with the phrase “two horse,” with two acting as an adjective on horse, though this is possibly not conducive to the overall process of language processing (no one speaks or writes that way) in that we would no longer be working with “natural” language, and may result in awkward constructions if working in an NLG context.

Now we would possibly want to remove stop words, those words or phrases that appear very commonly in the source tongue (in English, words like “and,” “is,” and “the” might be among them). I specify that we may only possibly want to do this as the removal of stop words can change the meaning of a sentence, and such removal could possibly alter the content of the sentence, but at the same time they may also only serve as linguistic “connectors” that serve no purpose beyond clarification to human listeners and readers.

The next step would be dependency parsing: this is essentially identifying more complex parts of a language, such as the primary noun of the sentence, prepositional phrases, and the like. Some of these linguistic structures are more obvious than others, but given that language has to be usable by its speakers, they have a tendency towards being repetitive in structure across a given language, which means they can be readily identified given sufficient data. While research in this area is ongoing, at time of this writing the spaCy package appears to be a solid implementation that is current and undergoing continuing development as research in this space continues.

From there, we can do named entity recognition: this is recognition of things like peoples’ names, geographic locations, dates, or names of events. This is an important step, as it is unlikely – for example – that “Korey” appears much in a corpus, if any, and is not technically part of the language: being able to identify proper names when doing NLP can be important. That said, however, it is entirely possible that a given data set will not have such information, and it would be dependent upon the sort of work being done whether or not this step is necessary.

The final step would be coreference resolution, in which terms like “it” are referenced back to the object of their reference. For instance, in “My horse is tired. She’s been walking all day,” the word “she” in the second sentence is a reference to “my horse,” and it would be useful for our analysis to reflect that. NeuralCoref is an extension to spaCy that performs this sort of analysis, and given that it is an extension of spaCy, that would most likely indicate that it, as well, is on the edge of research and development in this space.