Termination Project, Professor Harold Lewis

Week 1 Report

Grady Kurpasi

[gkurpasi@hotmail.com](mailto:gkurpasi@hotmail.com) (213) 304-8541

|  |  |  |
| --- | --- | --- |
| A R T I C L E I N F O  3 May 2020  Keywords: NoSQL, Document Databases, Graph Databases |  | A B S T R A C T  This week I investigated the storage needs of my project. I want persistent, flexible data structures but do not desire to code data handling myself. I had originally thought of traditional relational databases, but some research quickly uncovered new database paradigms that fit the data I intend to work with. I settled on using both Graph Databases and Document Databases. Next week I will be investigating a development platform that supports / unifies access to these two databases. I am initially considering a cloud implementation, but have to research capabilities and cost. |

1. The ‘Laboratory’

The ultimate goal of this project is to investigate some ideas about Word Sense Disambiguation (WSD) I have been building over the last two years. To facilitate this, the near term goal of this project is to construct a WSD ‘laboratory’, i.e. a development environment that facilitates the following:

1. Create a structured development environment that standardizes access to resources (e.g. corpora, existing machine learning algorithms)
2. Facilitate experimentation through varying parameters and algorithms
3. Standardize performance metrics
4. Provide persistent storage to facilitate comparison and aggregation of results and learning vectors.

The first thing I considered were storage requirements. Do to a background in relational databases (DB), I’m accustomed to considering the DB backend first and building application layers up from there.

Even cursory research quickly revealed that database technology has come a long way since my experience with relational database management systems (RDBMs). A wide array of NoSQL databases have since evolved. These databases are purpose built to handle non-relational, non-normal, unstructured data. More so, they go hand in hand with development paradigms and application architectures that are far more flexible and powerful than traditional RDBMs systems.

1. Storage Requirements

In the interests of just plain getting started. I did a rather informal requirements review to facilitate broad brush direction in researching technology. I considered the type of data that I’ve been working with through past WSD projects and my intuition for how I want to work with results.

* 1. Document Databases

First and foremost I considered the corpora that I’ve been working with. To enable Laboratory Goal 1 – Create a structured development environment and Goal 3 Standardize performance metrics, I knew that immediately I’d have to look at somehow standardizing sense inventories across corpora. One of the biggest drawbacks to comparing WSD efficiency between algorithms is differences in sense inventories and encoding between copora employed in various research (Raganato, Camacho-Collados, & Navigli, 2017). To approach this the laboratory will have to attempt some kind of normalization between tagged corpora. To do that I’ll need to easily read write and compare tagged corpora.

Tagged corpora are texts where each word is tagged with syntactic and semantic information. Different word types (adjectives, nouns, verbs, function words) may have different types of tags. Therefore data is both sequential and unstructured. Data of this type lends itself to encoding in JavaScript Object Notation (JSON)-like encodings. Therefore Document Databases are suited to storing large amounts of such data. This makes Document Databases an initial target for further research.

At first glance, it appears that a document database will make working with corpora much easier. Importing the Brown corpora (or a subset of it) will be an initial development task. That will allow me to investigate normalizing sense inventories as well as utilizing syntactic and semantic information as Recurrent Neural Network (RNN) input. When I look at WSD algorithms later, I want to use syntactic information to inform RNNs so that instead of just seeing a sequence of words, the RNN is informed of what words co-occur within phrase / sentence / paragraph boundaries.

* 1. Graph Databases

In addition to utilizing syntactic information in WSD, I also want to revisit using semantic input to further hybridize RNNs so that they benefit from Knowledge Based input. My ideas on how to do this are far less well formed at the moment, but I know I need a flexible data structure to standardize, possibly normalize input from WordNet and WordNet-like semantic resources.

Graph databases are optimized to represent relationships between data nodes or entities. Both nodes and edges (relationships) can have properties and because of the way they are stored with pointers, traversing graphs and retrieving both nodes and edges if very fast. Graph databases are the second topic of base research I will carry out.

* 1. Implementation

I know I will be developing in Python 3, so I’m looking to select Graph and Document database implementations. I’m also looking at related developments in application design to make initial architecture decisions about the project.

* 1. Cloud Computing

Right now, instead of coding a lot of infrastructure myself, I’m considering a ‘serverless’ Cloud implementation. This is a virtually hosted solution that abstracts several traditional layers of infrastructure supposedly allowing developers to focus on application logic. This might save me development time, execution time and possibly even expense in running long computations.

1. Next Week

I’ll being looking at cloud computing solutions. The leading solutions are Amazon Web Services and Microsoft Azure. My evaluation criteria will be:

1. Potential to support my project
2. Cost
3. Learning Curve

I’m new to all these technologies so development / design are going to be a fairly conspicuous feedback loop. My goal at any given point is to learn enough to avoid complete paradigm shifts later in development (wish me luck!) while keeping development moving forward.