Bilkent University

Department of Computer Engineering

Senior Design Project

Etymøn: A Deep-Learning Application for Etymological Clustering of Words

High-Level Design Report

Nashiha Ahmed, Mert İnan, Cholpon Mambetova, Utku Uçkun

Supervisor: Prof. Mehmet Koyutürk
Jury Members: Prof. Uğur Doğrusöz and Prof. Varol Akman

High-Level Design Report
Dec 24, 2017

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

High Level Design Report

Etymøn: A Deep-Learning Application for Etymological Clustering of Words

1. Introduction

Etymøn is an analysis and tracing tool for word origins in all languages. It will be used to review current etymological language families and if possible find new connections that were not already present in current taxonomy. It will accomplish this using a deep learning approach.

In the following sections, a brief description of the system and the system requirements are discussed. In addition, Etymøn's high-level design is also detailed.

1.1. Purpose of the System

Current etymological analyses rely on pattern matching or tracings between different languages by experts in linguistics [1], yet it may be cumbersome or even improbable to detect word origins in situations where direct links cannot be observed between two different words. In this case, Etymøn will pose an advantage as it will be using a large corpus of data in order to match words in any given language.

Various studies carried out by linguistic experts and historians improved the understanding of language and its origins [2]. However, there is still "room for improvement" in the field. Currently, most of the studies target the Proto-Indo-European language family [2]. There is sparse research done for other languages, and there is not a single, unified resource for this information. Most of the information is scattered online or among other forms of literature.

Since there is no similar project in the market yet, our software will be designed from scratch, which would make it a greenfield project. However, we will use other existing algorithms to build our software, such as deep learning algorithms among others that will be specified further in the report.

1.2. Design Goals

Design goals of the system in the areas of performance, dependability, maintenance and end-user criteria are described below/

Performance:

- Response Time: Etymon should be able to respond to user's queries within an unnoticeable delay. After finding the result of the user's query, it should be able to zoom into to the map smoothly and within unnoticeable delay.
- Throughput: As there can be multiple users of the system at the same time, Etymon should be able to respond to all of the search queries of all the users. The system should perform search and map traversal functions for each user at one time.
- Memory: Space required for Etymon will be entirely dependent on the size of the data. Machine learning model and codes of other modules will not occupy space as the data. Data to train should be compressible and it should not occupy space on the devices of the users, hence an online server is required to store all the training and testing data and the generated language sea.

Dependability:

- Robustness: Users may input invalid search words. Etymon should be able to survive invalid user inputs.
- Reliability: Etymon's machine learning components should behave as described. Differences between observed and specified behaviors of the system will be tolerated minimally.
- Availability: The system should be available all the time as the queries can be received at all times. Down times of servers should be minimal.

Maintenance:

- Extensibility: Etymon should be extensible in its machine learning, augmented reality and object recognition modules. As these modules can be updated in the community quite rapidly, the system should be able to adapt to changes in those modules and allow interfacing of new classes and models.
- Modifiability: Especially in terms of its machine learning algorithm, Etymon should be able to allow change. As better machine learning models can be developed, Etymon should be able to accommodate these changes and train the model again on the data.
- Readability: As the code of the system will be minimal yet math-intensive, the code should be readable.
- Traceability of requirements: It should be easy to map the code of specific subsystems to specific requirements.

End-user:

- Utility: Etymon should help the user to understand the connections between different languages and help the user's work in comprehending the connections between different words and their origins.
- Usability: It should be very simple for the user to traverse the system. Using Etymon should not require any tutorials or training.
- Understandability: Understanding the working mechanism of Etymon should be easy. Tracing the correctness of the results by the users should be possible.

1.3. Trade-Offs

Overfitting vs. Underfitting: During the training of the machine learning algorithm, overfitting will be favored against underfitting. As the training data is the only aspect that is important in this report, underfitting cannot be tolerated.
Precision vs. Accuracy: Even though both precision and accuracy of the results are favored, if a choice will be needed, then precision will be favored. As the accuracy of the mapped words cannot be checked for their correctness, precision among different language families is more important than the accuracy of classification of that word.
Delivery Time vs. Functionality: If testing and training runs behind the schedule, leaving out features such as augmented reality or hallucination may be possible, even though undesirable.
Model Complexity vs. Machine Type: As there are multiple machines with different efficiencies that can run machine learning algorithms, they will be put to use. Instead of increasing model complexity, in order to gain faster results, machine type will be changed to a faster one.

2. Proposed Software Architecture

2.1. Overview

In this section of the report, we will be analyzing some components of the high level architecture of Etymøn. Subsystem decomposition will be presented with its UML diagram in order to simply show the modules in the application domain to the problem.

2.2. Subsystem Decomposition

An ideal candidate architecture for our purposes and that fits the design of Etymøn is the client-server architecture. The user's machine will serve as client and will request and receive queries from a machine learning server. The server is our host computer where permanent data will be stored and processed, such as the etymological map, connections among words and languages; whereas the client side may perform some tasks such as animation, AR and word hallucination.

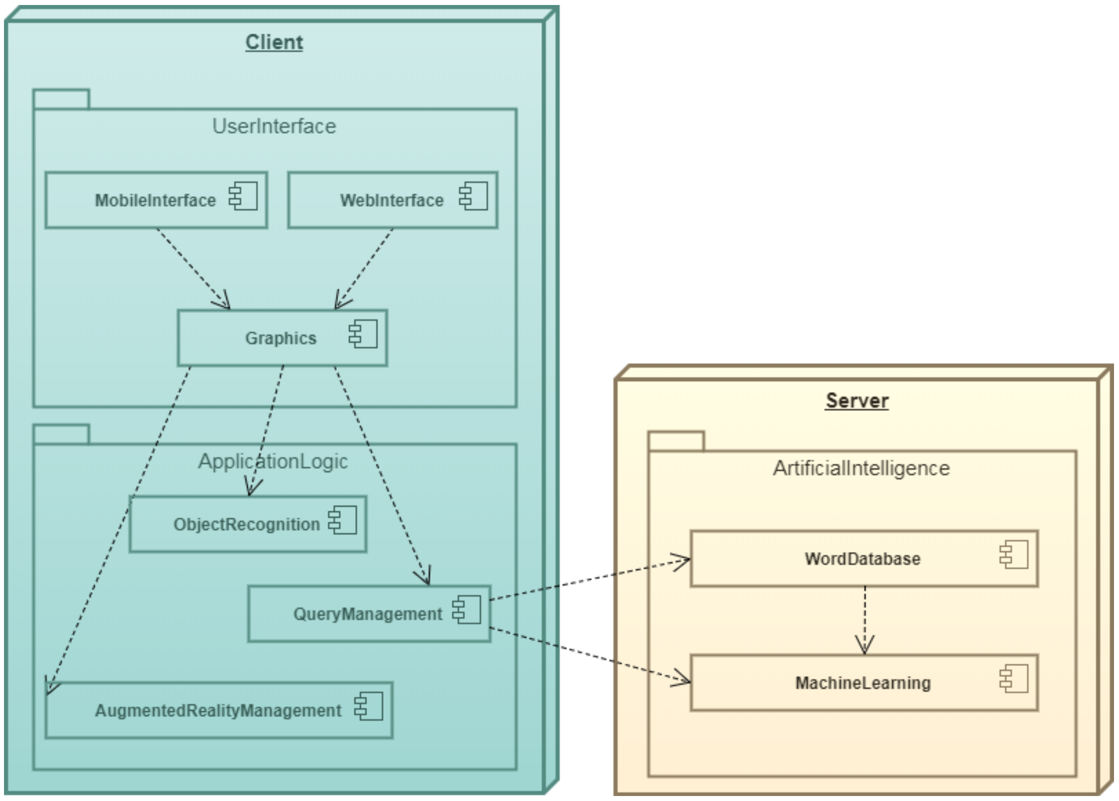

Figure 1: This figure shows the subsystem decomposition of Etymon and the architectural design.

2.3. Hardware/Software Mapping

Since we want Etymøn to work for all languages and deliver a fast word look-up service, it will require good hardware to run on. Storing words, their pronunciations and meanings for multiple languages will require great amount of memory space. Also, running deep learning algorithms on this graph and traversing it in a reasonable time will require computational power. Therefore, we must use a supercomputer to run our program in addition to our simple desktop computers. For this, we will use Google's Cloud Machine Learning Engine [3], TensorFlow [4]. Smartphones will also be required for the augmented reality and object recognition components of the project.

2.4. Persistent Data Management

Etymøn's effectiveness highly depends on its data management. It will find, store, manage and present information. For this reason we decided to use a robust database system which allow us to store excessive amount of data as quickly as possible. Etymøn will store many words from many different languages. For each word it will also store its meaning, pronunciation but also the data which will be used in deep learning process.

The word and pronunciation data will be acquired from Wikipedia. Furthermore, for English, pronunciation data from online sound libraries will also be used, and for other languages, International Phonetic Alphabet (IPA) readings of the words will replace sound files for clustering. To store meanings of the words, online dictionaries can be used. Using this data, we will create a Word Database.

2.5. Access Control and Security

Our software has no authentication method and can be accessed by all users. All users have the same privileges except for the admins. The end-users of Etymøn will not have direct access to word database of the program. They will be authorized to traverse the word information freely but will not be able to alter it. Administrative users will have access to graph modification tools. They will be able to alter the data of words, alter the relations formed by deep learning algorithm, add or remove words etc. The program will run on procedure-driven centralized control for the control of the software itself with triggers from the user, and the control will reside in controller objects in the program code.

2.6. Boundary Conditions

In the boundary conditions, the system will act according to the following.

- Initialization: When Etymøn is brought from a non-initialized state—such as the initial executable of the program—to the steady-state, the main processes will be done by the user interface subsystem. The user interface will display the splash screen during startup and then show the Language Sea.
- Termination: All cleanup procedures will be started by the program controller. No single subsystem is allowed to terminate on its own. Every subsystem is notified if a single subsystem is terminated. Read and write to the filesystem will be completed before termination.
- Failure: If a failure occurs during read and write to the Word Database or search, warning messages will be show. If failure occurs during getting the background images or component images, a restart of the software will be requested.

3. Subsystem Services

The services that the subsystems will provide are on the interfaces of client and server. As communication is necessary between them, information providers between modules such as etymological data will be sent through these services of the subsystems. These services are given in Figure 2.

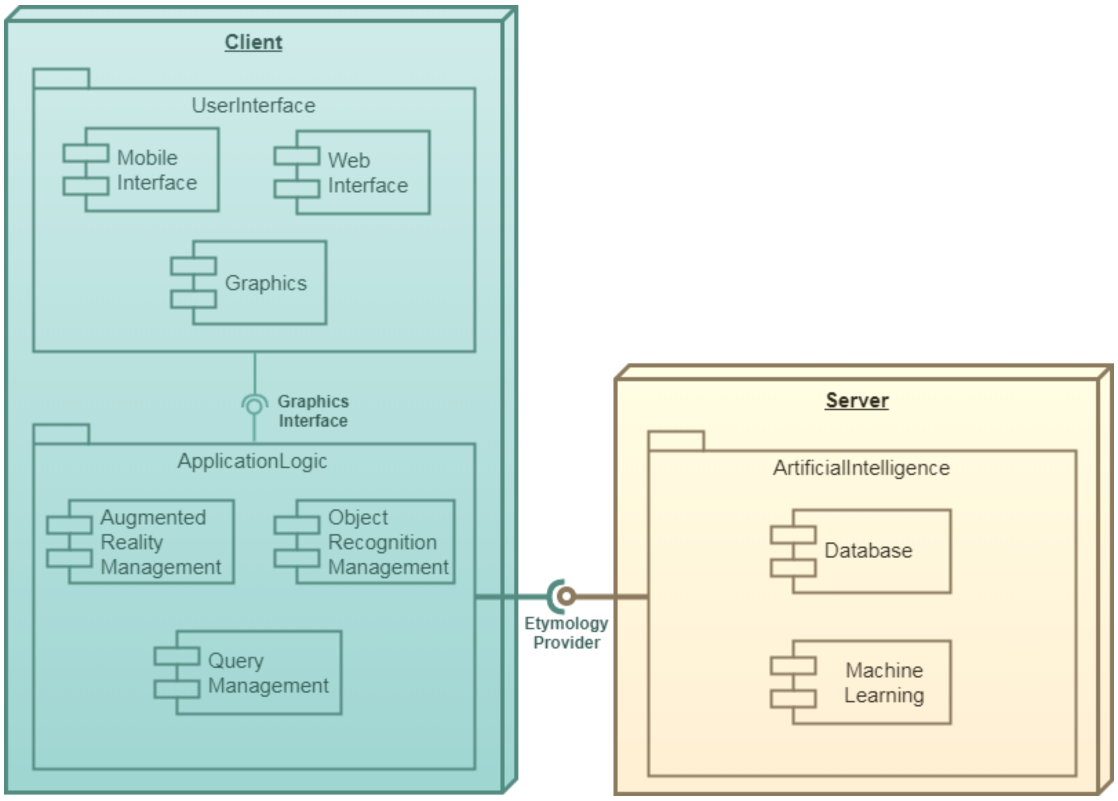

Figure 2: This figure shows the subsystem services of Etymon.

4. References

[1] C. Diagne and N. Barradeau, *Free Fall,* https://artsexperiments.withgoogle.com/freefall/wave. [Accessed: 09-Oct-2017].

[2] J. Redmon, *YOLO: Real-Time Object Detection*. [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed: 09-Oct-2017].

[3] "Cloud ML Engine Overview | Cloud Machine Learning Engine (Cloud ML Engine) | Google Cloud Platform," Google. [Online]. Available: https://cloud.google.com/ml-engine/docs/technical-overview. [Accessed: 25-Dec-2017].

[4] "TensorFlow," TensorFlow. [Online]. Available: https://www.tensorflow.org/. [Accessed: 25-Dec-2017].