Vocabulary learning with in the context of user domain

Haemanth Santhi Ponnusamy
hameanthsp@gmail.com

Himanshu Bansal
email@domain

Abstract

We propose a vocabulary learning application targeted on the set of user who already posses some basic skills in a language and bored of the traditional method of learning words in the language. Our design efficiently structure the vocabulary space and track the user performance. It also provides a great advantage in auto-generating the content for the entire learning process with almost zero human effort. We allow the user to choose the text of there own interest. Then all the activities and feedbacks are generated only from the chosen text.

1 Introduction

Vocabulary learning is an open ended task as the languages are vast and still evolving with the addition of new words now and then. This makes it hard for the user to get a sense of progress. Many of the vocabulary learning application starts from very basic words in the language. Which makes more sense for the beginners, but not for our target users. They might want to start at an intermediate level. Also the example sentences used in these tools as a context of a word are mostly very generic and some time unnatural. This nature is due to the fact that those application are designed to support users irrespective of their language competence, background, area of interest, learning goals. Also It is hard for the developers to manually create content to satisfy all kind of users.

In this paper we propose an approach to overcome all the above mentioned difficulties to build an application that could potentially satisfy our targeted users. We do this by partially sharing the problem with the user to choose the text of their interest and reading level. Then we process the chosen text to select the candidate words and build a network of candidates to efficiently model the vocabulary of the text and generate useful activities from them. This help the user in defining the definite space to master. By this way one could choose to learn words used in a specific domain with specific context.

2 Related works

Details of some of the existing tools goes here.

3 Our application

3.1 Building vocabulary

The raw input text from the user is structured into a meaningful form that is convenient for vocabulary learning. As we use some statistical feature to process the user text, the quality of structured data is directly proportional to the size of the user text. The process of building the vocabulary with respect to the user text is independent of the user. We focus to capture the entire vocabulary structure of the text. Thus any new user can selc

3.1.1 Candidates

All the words in the language are not equally important. Given that the users already know some basics of the language, we could eliminate the most frequent, functional words (the, of, in, on etc...), rare/very less frequent words and improperly parsed words. So the words occurring below the frequency of 10 and the stop words(https://github.com/explosion/spaCy/blob/master/spacy/lang/en/stop_words.py) are eliminated to handle the above cases.

The words that occur in different parts of the speech could posses different sense. In-order to differentiate between them. We represent each candidate word as a pair of word and its POS tag.

(word, POS tag)

3.1.2 Sentences

All the sentences in which the candidate words occur are cleaned and mapped to the corresponding candidate word. So each candidate word is mapped to all the sentences in which it occur and also each sentence is mapped to all the candidate words it contains.

3.1.3 Complexity

There are lot of methods to measure the complexity of a word. We choose frequency. In a language, we consider the words that are less frequently used are as more complex / uncommon and the words that occurs more frequent are as easier / well known. We use the frequencies obtained from SUBTLEX-US Brysbaert and New (2009), a database of 50 million words from various English-US movies and TV series subtitles.

$$C_w = \frac{1}{\log_{10}(freq_w)} \tag{1}$$

Where $freq_w$ is the average frequency of word w per million words in the database.

3.1.4 Vector representation

Also each of the candidate is mapped with a semantic vector representation which is obtained from the pre trained model *en_web_core_lg* of *spaCy* (https://spacy.io/), a natural language processing library. The main drawback is that it cannot address the out of vocabulary(OOV) words. Which could be rectified by training a custom word vectors over a decently sized user content.

3.2 Creating structure

3.2.1 Family

We group the words into families, similar to Bauer and Nation (1993) work on word families but instead of seven sub-groups we form a single group for all the types. The main intuition of grouping the words into families is that the user can extrapolate their knowledge of inflections of a language to understand/predict all the possible forms of an unseen word. Similarly with this setup our system can extrapolate the mastery of one the member to the entire family. Which could drastically reduce the number of interaction that the system need to estimate the user's vocabulary.

3.2.2 Network

Now the families has to co-exist in the space of language(limited by the user text) as a network like a society. It is a fully connected network with each family with a different affinity to another. The affinity is a measure of contextual similarity between the families. We compute cosine similarity between the mean of word vectors of all the members of the family to similar mean vector representation of another family.

$$V_{F_i} = \frac{1}{n}\sum_{k}^{n} V_{w_k} \tag{2}$$

where n is the number of elements of the family F_i.

$$S_{ij} = \frac{V_{F_i} * V_{F_j}}{\|V_{F_i}\|\|V_{F_j}\|} \tag{3}$$

where, S_{ij} is the cosine similarity between the vector representations of families F_i and F_j.

As we can see in the figure 1 the neighbors of the terms in the graph are tightly bound with the neighbors that are contextually closer.

By this way we create more structure in the space of vocabulary. Which come handy in many situations like activity creation, updating mastery of each vocabulary and analysis. This structure helps in further reducing the search space by allowing the model to get a better inference about learners level with relatively very less and effective interactions compared to a method of tracking each word in the vocabulary individually.

3.3 Content organization

3.3.1 Books

All the processed data such as vocabulary, families, network, sample sentences are packed into a book instance. This also tracks the meta information such as Title, Author, Genre, Year and Publisher. This help any future user to select the processed book directly. This also could help in comparing the performance of different users on the same book.

3.3.2 Bookshelf

All such processed book are organized in multiple bookshelves specific to each domain similar to the gutenberg project (http://www.gutenberg.org).

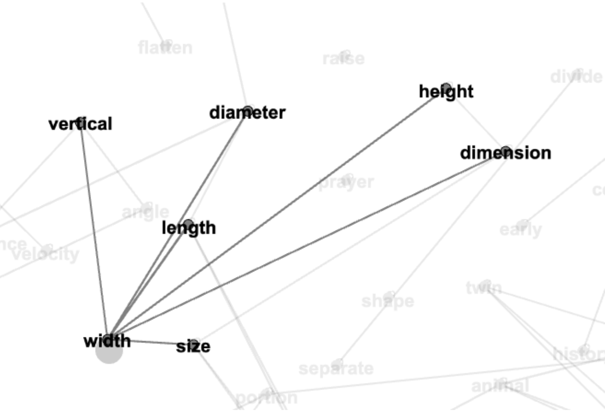

Figure 1: neighbors of 'width'

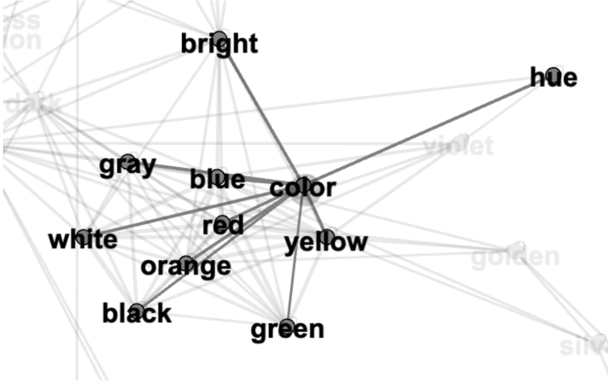

Figure 2: neighbors of 'color'

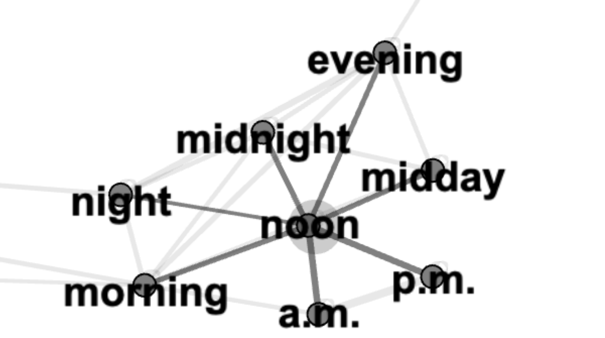

Figure 3: neighbors of 'noon'

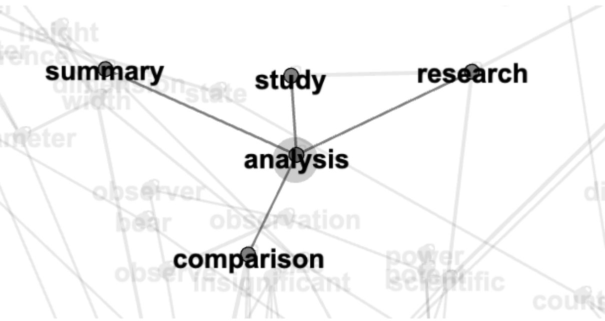

Figure 4: neighbors of 'analysis'

3.4 Models

In this work we maintain multiple models to track and update different aspect of the application.

3.4.1 Learner

The learner instance maintain the personal information of the user and tracks the list of instance of books the user has choose to improve vocabulary and the progress in them. This also could maintain overall vocabulary knowledge of the user and customize the activity type, feedback and book suggestions based on individual needs.

3.4.2 Tutor

For each book the user selects, a tutor instance will be created. The main activities of the tutor are to design a learning session, evaluate the performance and track the mastery level of the user w.r.t all the vocabulary in the book. Then again generate a new session based on the updated mastery levels in the network.

Mastery Score: The value ranges between 0 and 1. Initially it is assigned to 0.5 to indicate the uncertainty. Based on the performance of the user it is either increased or decreased by a factor. This approach implicitly capture the un-visited nodes in the network.

Update rule: As we have built a network of families capturing the contextual similarity. We can incorporate this into our update rule to update the mastery scores. When we get some outcome for an activity involving a member from the family F_i.

$$M_j = M_j * (1 + (\alpha * sign * S_{ij})) \qquad (4)$$

Where M_i is the mastery of the family F_i. α is a tunable parameter for the magnitude of an update. $sign \epsilon \{-1, +1\}$ is the direction of the update. It depends on the correctness of user response to the corresponding activity. And S_{ij} is the measure of contextual similarity between the two families F_i and F_j.

3.4.3 Session

An instance of session is created by the tutor. Which decided list of word families to be practiced. The key functions of this model are to deciding the interaction type (teaching, testing, feedback), compose an interaction with all required data and handles the flow and closure of the session.

In order to make the learning more efficient the most critical nodes of the graph are selected for a session. Here, The criticality of a node is decided based on the intrinsic(complexity) and extrinsic(degree, quality of connection) nature of the node.

3.5 Activity

Since our motive is to reduce the effort for content creation. We generate the activities on the fly. In this work we generate two type of activities.

fit to context: The user is prompted to complete 3-4 incomplete sentences with one among the given list of word suggestions. The options are chosen to be contextually tight to improve the quality of the activity and user learning outcome.

scrambled word: The user is prompted to come up with a word from the set of characters to complete an incomplete sentence. The words with word length less than 6 characters are allowed to generate this activity. Since the larger answer words makes it more ambiguous for the user to solve.

Currently, the activity types are chosen in random for the answer words of length more than 6 characters. This could also be enhanced to chose based on the user interactions.

3.6 Distractor selection

The distractors plays a important role in deciding the quality of the activities. We take advantage of the network of families we built based on the contextual closeness to overcome this problem. Here we rank the neighbors of the answer family and chose a best set below a threshold to avoid the synonyms. From the best set of families we choose the members which matches the POS tag of the answer word to make all the distractors coherent.

3.7 Implementation

4 Future Works

The system has limitation on operating on the words that does not have a learned representation on the vector space. Which could be rectified by learning a custom word vector specific to the domain. This feature could create new use case like a tool to learn jargon specifics to a new domain.

Teaching the word in a session. Control over the complexity (tunable complexity) Tracking words at learner level across books (to have a warm start for new book)

Acknowledgments

References

Laurie Bauer and Paul Nation. 1993. Word families. *International journal of Lexicography*, 6(4):253–279.

Marc Brysbaert and Boris New. 2009. Moving beyond kučera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior research methods*, 41(4):977–990.