Using Recommender System Techniques to Predict the Words a Child Will Learn Based on Current Vocabulary

Kelly Macdonald
Westmont College
kmacdonald@westmont.edu

Professor Eliana Colunga
University of Colorado, Boulder
colunga@colorado.edu

Introduction

Recommender systems, which use machine learning algorithms to solve problems such as recommending content or a product to a user, are widely used for commercial purposes. For example, Netflix uses a user's preferences to recommend movies the user has not yet seen but might enjoy, and Amazon uses past purchases and item viewings to recommend items that a user might like to buy in the future [1].

For these experiments, several of these same algorithms and techniques are applied to a new problem: predicting the words a young child will learn next based on the words that child knows now. Experiments on the vocabularies of young children often require the assistance of a parent or guardian to indicate which words the child already knows, and being able to recommend new words to the parent might make it easier to gather this information. In addition, it is hoped that algorithms such as these might reveal patterns in children's vocabularies and increase understanding of the speech learning process. As more is learned about the ways in which humans acquire speech, it becomes easier to identify atypical learning patterns, which in turn can help diagnose problematic learning patterns.

Recommender system techniques generally deal with matching users to items. For these experiments, the users become children and

the items become words. Recommender system algorithms tend to take either a user-based or item-based approach [1]. A user-based approach uses the interests (or in this case, knowledge) of similar users to recommend items to the given user. An item-based approach looks at items similar to the items already associated with the given user and attempts to find similar items to recommend. Both of these approaches are taken here: the k-NN and American CDI norm algorithms described take a user-based approach, while the probabilistic, SVD, and noun/attribute algorithms can all be (at least loosely) classified as item-based approaches. However, the "straw man" algorithms to which these algorithms are compared do not quite fit into either category.

Data

Three data sets were used for these experiments: the child/word set, the noun/attribute set, and the American CDI norms set.

Child/Word Set

The child/word set contains cross-sectional data on 100 children and their knowledge of 677 words, in the form of a matrix (with a 1 indicating that the child knows the word and a 0 otherwise). In addition, the child set contains information such as each child's gender, age in months, and percentile (how the child compares with other children at his/her age based on the number of words he/she knows). Vocabulary sizes range from 5 to 661. Ages range from 12 to 32 months. This data set was used for all of the methods described here, though not all of the information was used for each method. The list of words included in this data is a subset of the Macarthur-Bates Communicative Development Inventory (MCDI) word list. For these experiments, 677 of these words were used.

Noun/Attribute Set

The noun data set contains information about how well each of the nouns fits into

each of 97 categories (or where they fall on a scale). Each noun has a score from 0 to 10 (inclusive) for each category (or attribute). To obtain these scores, a group of adults was asked to score each noun in each category, and the average of each of these score sets was used as the final score for a given word and attribute. The list of nouns is separate from the list of words in the child/word set, which contains words of many parts of speech. There are 327 words that are featured in both of these lists, and only these words were used for algorithms that utilized the noun/attribute set. This data set was only used for some of the methods described here, and when it was used, all non-nouns were excluded from the experiment. If a child's vocabulary contained fewer than two nouns, the child was excluded from the experiment.

American CDI Norms Set

The American Communicative Development Inventory (CDI) norms set [2] contains information about the percentage of children who know each word in the set at each age from 16 to 30 months (incrementing one month at a time). There is one set of these norms for female children, another for male children, and one for both combined. This data set contains all of the words included in the child/word set. It was gathered from 1461 children.

Measuring Accuracy

To test the accuracy of each of these algorithms, one fifth of each child's vocabulary was removed, and the algorithm was used to try to recommend it back. Once a list of recommended words had been compiled, the number of correct recommendations was divided by the total number of recommendations to get the percent correct. Each algorithm was run 100 times, and the average accuracy rate and variance over all of these runs was recorded. The exception to this is the SVD algorithm; this algorithm was run multiple times, but

the results of only one iteration were recorded. When an algorithm required information, such as probabilities, to be gathered from the data before recommendations were made, this information was gathered only from the 80% of the child's vocabulary that was not taken out, in order to realistically simulate a real-world recommendation scenario.

The constraints of the experiment had some effects on the results that must be taken into account prior to analysis. Because only a few hundred words were used, and some children knew nearly all of those words, there was little guessing involved in making correct recommendations for those children with large vocabularies. If very few words are not in a child's vocabulary, then there are very few possible wrong recommendations, and the algorithm will always recommend all or most of the words correctly.

On the other hand, if a child does not know many words, there will be very little to base predictions on (though this does not matter for these "straw man" algorithms), and there will be hundreds of possible wrong answers and few correct ones. For a child who knows only 5 words and has one of those words removed for the purpose of recommending it back, the algorithm will either predict that one word correctly and have 100% accuracy, or predict incorrectly and have 0% accuracy. The latter case is much more probable. Unlike the somewhat unrealistically high accuracy found in recommendations for children with large vocabularies, this trend represents a real-world recommendation scenario fairly accurately.

"Straw Man" Algorithms

To identify the lower bounds of success, the accuracy and variance of two very simple "straw man" algorithms were measured: one of these algorithms simply recommended

random words for each child, and the other calculated which words were most commonly known to all of the children and recommended the most commonly known words that each child did not already know. The first of these algorithms had an average success rate of 21.5%, while the second had an average success rate of 52.0%. The average accuracy of both algorithms increased as vocabularies got larger, while the variance decreased. This happened somewhat more gradually for the second algorithm than for the first.

While both algorithms produce a higher variance in results for children with small vocabularies than for children with large ones, the second algorithm has a much higher variance for small vocabularies because the correct words were actually guessed more frequently. This is strikingly different from the random recommendations, which had 0% accuracy and 0 variance for the smallest vocabularies before the trend of high variance appeared. It is realistic to assume that these trends in variance would exist in a real-world recommendation scenario involving just about any number of vocabulary words.

K-NN

K-NN approaches are commonly used for recommender systems because they are inherently related to the recommender task of identifying like-minded users [1]. Three different k-NN algorithms were implemented with the child/word set.

The first k-NN algorithm simply assigned each child a similarity score based on the number of words they had in common with the given child. Then, each word received a score equal to the number of top K most similar children who knew it, and the words with the highest scores were recommended. Several values of K were tried, and the value of K that scored highest was selected and recorded. This algorithm is equivalent to the

second "straw man" algorithm, recommending the most popular words in overall, if K is equal to the total number of users, but it can theoretically give different results for lower values. However, running this algorithm several times revealed that the percentage of accuracy was highest for values of K very close the number of users, and no one particular value of K stood out. Furthermore, the percentage of accuracy was not statistically significantly different from that of the algorithm which recommended the most popular words overall. Although this algorithm appears to be superior to the first "straw man" algorithm, it is at best exactly the same as the second.

The second algorithm was similar to the first, but each word received a score that was equal to the sum of the scores of the top K children who knew it, and the words with the highest scores were recommended. This gives more weight to words that are known by children more similar to the current child, and it is not equivalent to the second "straw man" algorithm. Several values of K were tried for this algorithm as well, and the value of K that scored highest was selected and recorded. This algorithm had an average accuracy rate of 50.5% for the best value of K. Interestingly, this algorithm also scored highest with values of K that were near or equal to the total number of users. Although it is not technically equivalent, it performed very similarly to the second straw man algorithm.

For the third algorithm, words were scored the same way as in the first, but each child's similarity score received a boost based on that child's other attributes. Binary attributes like gender, language, number of languages spoken, and hardness of hearing were worth equal boosts if a child's value for that attribute matched the given child's value. For attributes like age, percentile, and words spoken, boost was proportional to the difference between the value for the current

child and the value for the child being scored, and the child's similarity score was multiplied by this value. K was selected the same way as in the first two algorithms. This algorithm had an average accuracy rate of 51.1% for the best value of K and, like the other k-NN algorithms, scored highest for very high values of K.

Patterns in average accuracy and variance for these algorithms were largely the same as the patterns found in the two "straw man" algorithms.

Probabilistic

Next, four probabilistic algorithms were implemented. This involved calculating the probability that a child knows each word based on each other word. (For example, the probability that a child knows "mommy" if that same child knows "daddy" was calculated.) These probabilities were calculated on a training set. When making recommendations, each word's score was calculated in a different way for each algorithm.

For the first algorithm, the score of each word was calculated by using the "inclusive or" probability formula to combine the probabilities that the child knew that word based on each word that the child already knew. This algorithm had an accuracy rate of only 26.5% - barely better than random, and far worse than recommending the most common words.

For second algorithm, the probabilities were instead combined as a simple sum. This algorithm was accurate 49.1% of the time, a huge improvement over the first, but still not enough to beat the second "straw man" algorithm.

The third algorithm used the same scoring method as the second algorithm, but also added the probabilities that the child knew the word based on the child's gender, age, percentile, and number of words spoken. It then recommended the words with the

highest scores. This algorithm had an accuracy rate of 52.5% percent, about the same as the second straw man algorithm (technically higher, but not enough to be considered a statistically significant improvement).

Finally, the fourth algorithm used a different method than the first three for determining the relationships between different words. The child/word matrix was multiplied its own transpose so that each row and column of the result represented a word, and each cell contained a number related to the number of times that the word for that row appeared in a child's vocabulary with the word from that column. These numbers were then summed to calculate word scores. The numbers gathered through this method amplify the relationships between words, and the algorithm is a lot more efficient than the first three. However, it was only accurate 39.9% of the time.

Interestingly, although average accuracy increased with vocabulary size, much like the "straw man" and k-NN algorithms, variance also increased with vocabulary size. This trend in variance is the exact opposite of the trends in variance displayed by the previously described algorithms.

Latent Semantic Analysis

Latent semantic analysis is a popular technique for recommender problems that involve processing text. For example, a search engine will use this technique to take a user's query and match it with relevant documents. The process involves singular value decomposition, a matrix decomposition method that draws out latent "concepts" from the data and gives both the query and each of the documents a score based on how much it aligns with each concept. For purpose of these experiments, each word was treated like a query, and a child's vocabulary was treated like a document. On this data set, just two latent

concepts were detected. The distance of each word from each child was measured as the cosine of the angle between the child vector (containing the child's score for each concept, based on his/her vocabulary) and the word vector (containing the word's score for each concept). The the idea is that this method should be able to pick up on subtle patterns in children's vocabularies, patterns which are not easily observable by simply looking at the words. However, this algorithm was only accurate 30.6% of the time.

The accuracy of these algorithms increased with vocabulary size, and so did the variance, as with the probabilistic algorithms.

Noun Features

Four algorithms made use of the noun/attribute set, which contained information about how well each noun fit categories like "is round" and "has face," or fell on scales such as small to large and light to heavy.

The first algorithm involved calculating the amount of each attribute that each child had in his/her vocabulary based on the sums of the attribute scores for each of the words that the child knew. The child's score for each attribute was then multiplied by each word's score for that attribute, and these values were summed across all the attributes to determine each word's score. This algorithm had an accuracy rate of 27.2%

The second algorithm was similar to the first, but the child's score was an average rather than a sum, and the cosine formula was used to determine each word's score. These algorithms had an accuracy rate of 25.1%.

The third algorithm was like the second one, but an entropy calculation was used to remove attributes that had little informational value Entropy levels ranged from 0.42 to 2.18; several different entropy

thresholds were tried before 2 was eventually settled on. Any attribute with an entropy level below this threshold was eliminated, leaving just 20 attributes. This algorithm was accurate only 24.1% of the time.

Finally, the fourth algorithm, which also used an entropy calculation, used averages to score words and users, and calculated each word's score as the cosine of the angle between the user and the word. This algorithm was accurate 24.3% of the time.

Although none of these algorithms did much better than random, an interesting pattern could be observed in recommendations that they produced, particularly in the recommendations for children with smaller, more homogenous vocabularies. A child who only knew the words "mommy," "daddy," and "baby" (with one of these words randomly being removed each time) consistently received other "people" words, like "aunt" and "grandpa," as recommendations. Likewise, a child who knew only animals received other animals as recommendations. The recommendations appeared to be intelligent on some level, just correct very often.

These algorithms returned to the trends demonstrated by the "straw man" and k-NN algorithms, with average accuracy increasing and variance decreasing as vocabulary size increased.

MCDI Norms

Finally, two algorithms were implemented using the American CDI norms data set. Because this data was gathered from a much larger group of children than the group used for these experiments, overfitting was not as much of a concern.

The first algorithm used the single set of norms for both male and female children. Any children younger than 16 months were assigned the norms for 16-month-olds, and any children older than 30 months were

assigned the norms for 30-month-olds, while all the other children were assigned the norms for their age. Scoring words was simple: once the norms for the child's age were extracted, the percentages were matched up with the corresponding words and sorted. The percentages were the words' scores, and the words with the highest scores were recommended. This algorithm had an accuracy rate of 47.7%.

The second algorithm made use of the separate male and female norms, so that the norms used for each child were based on both age and gender. Scoring worked the same way as for the first algorithm, and this algorithm was accurate 47.3% of the time.

For these algorithms, average accuracy increased and variance decreased as vocabulary size increased.

Discussion and Future Work

Although all of the algorithms described here did better than random, and many came close to the success of recommending the most popular words (even essentially tying it), none could actually beat the second "straw man" algorithm. However, many of the algorithms brought out interesting patterns in the children's vocabularies. Some algorithms and data sets were more suited to certain vocabularies than others. For example, using the combined gender American CDI norms resulted in a higher accuracy rate than using separate ones for 54 children, while 41 did better with separate norms and 3 were tied. Although the difference in accuracy was often negligibly small, several children displayed statistically significant differences between the two algorithms – for one child, the combined norms were yielded a 52% accuracy rate, while the separate norms were accurate only 37% of the time. The two algorithms with the highest overall performance – the one recommending the highest words (the second "straw man" algorithm), and the

probabilistic one which added the probabilities based on attributes outside of vocabulary – did not always have the highest performance for every individual child.

The data sets used for these experiments contain several pieces of information in addition to the words in each child's vocabulary, and figuring out how to combine these pieces of information with the set of vocabulary words is no trivial task. Many of the algorithms here used relatively simplistic techniques for utilizing this additional information (if it was utilized at all), and future work may involve using more complex methods to determine the appropriate weights and the relationships between these attributes and the words that the child will learn. It is also likely that there are factors not included in this data which influence the development of a child's vocabulary, and while some of these may be possible to gather and organize into easy-to-use data sets, others may be more difficult to grasp.

Furthermore, the algorithms used here are only a small subset of all the techniques that have been studied and used for recommender systems. Among those that are not described here are Bayesian belief networks, artificial neural networks, and probabilistic latent semantic analysis (PLSA), to name just a few popular approaches. Bayesian belief networks are a somewhat more complex approach to dealing with probabilities than the probabilistic algorithms discussed here. Artificial neural networks have been known to perform similarly to Bayesian belief networks, but are very different in implementation and do not rely on linear relationships between probabilities. Finally, PLSA, which is closely related to LSA [3], has been known to outperform LSA, but is has a high complexity and can be difficult to implement efficiently. While an efficient

Ruby gem was found and used for the LSA implementation described here, similar tools for PLSA are difficult to find and may have to be built from scratch. Each of these algorithms brings out different aspects of the data in vastly different ways and would definitely be worth looking into further in the future.

References

1. Ricci, Francesco, et al. Recommender Systems Handbook. Springer, New York, 2011.
2. Dale, P. S., and Fenson, L. (1996). Lexical development norms for young children. Behavioral Research Methods, Instruments, & Computers, 28, 125-127.
3. Bhaskar Mehta, Thomas Hofmann, and Wolfgang Nejdl. 2007. Robust collaborative filtering. In*Proceedings of the 2007 ACM conference on Recommender systems* (RecSys '07). ACM, New York, NY, USA, 49-56. DOI=10.1145/1297231.1297240 http://doi.acm.org/10.1145/1297231.1297240
4. Larry Fenson, Philip S. Dale, J. Steven Reznick, Elizabeth Bates, Donna J. Thal, Stephen J. Pethick, Michael Tomasello, Carolyn B. Mervis and Joan Stiles. Monographs of the Society for Research in Child Development, Vol. 59, No. 5, Variability in Early Communicative Development (1994), pp. 134-139