Using Collaborative Filtering to Model Argument Selection

Sagar Indurkhya
Massachusetts Institute of Technology, Cambridge MA 02139, USA
indurks@mit.edu
&
Virtualitics, Inc

Abstract

This study evaluates whether model-based Collaborative Filtering (CF) algorithms, which have been extensively studied and widely used to build recommender systems, can be used to predict which common nouns a predicate can take as its complement. We find that, when trained on verb-noun co-occurrence data drawn from the Corpus of Contemporary American-English (COCA), two popular model-based CF algorithms, Singular Value Decomposition and Non-negative Matrix Factorization, perform well on this task, each achieving an AUROC of at least 0.89 and surpassing several different baselines. We then show that the embedding-vectors for verbs and nouns learned by the two CF models can be quantized (via application of k-means clustering) with minimal loss of performance on the prediction task while only using a small number of verb and noun clusters (relative to the number of distinct verbs and nouns). Finally we evaluate the alignment between the quantized embedding vectors for verbs and the Levin verb classes, finding that the alignment surpassed several randomized baselines. We conclude by discussing how model-based CF algorithms might be applied to learning restrictions on constituent selection between various lexical categories and how these (learned) models could then be used to augment a (rule-based) constituency grammar.

1 Introduction

In learning a language, a child solves many difficult puzzles using limited input data (Berwick et al., 2011; Piattelli-Palmarini and Berwick, 2012; Lasnik and Lidz, 2017). One such puzzle involves the child deciding whether a given verb can take a particular noun as its complement when the child has never previously observed that verb and that noun co-occur in a sentence. To illustrate this puzzle, let us consider an example - suppose the child learner has heard the following sentences:

(a) *"I lied and and said that I would not **smash** the **windshield**."*
(b) *"The robber was not planning to **smash** every **plate**."*
(c) *"I want to take this hammer and **smash** the precious **vase**!"*
(d) *"They are going to **shatter** the **windshield**!"*
(e) *"The boy who was busy staring at his phone will trip over and **shatter** his mother's favorite **plate**."*
(f) *"Did you **shatter** the blue **vase**?"*
(g) *"She knew that Susan would **break** her expensive new **windshield**."*
(h) *"I saw the man in the red sweater **break** the delicate **plate**."*

Now suppose that the child learner has never heard a sentence in which the verb "break" takes the noun "vase" as its complement. How should the child decide whether the following production is licit?

(i) *"He is going to trip over and **break** the **vase**!"*

One strategy that the child might employ is as follows. First the child observes that the verbs "smash" and "shatter" behave similarly by noting that both verbs can select any of the three nouns "windshield", "plate" and "vase" as a complement (see sentences (a-c) for "smash" and sentences (d-f) for "shatter"). Then the child observes that the verb "break" appears to be similar to the verbs "smash" and "shatter" by noting that the three verbs can select "windshield" and "plate" as complements (see sentences (a, d, g) for "windshield" and sentences (b, e, h) for "plate"). On the basis of these observations, the child may decide that if the verb "break" really is similar (semantically) to "smash" and "shatter", then "break" should also be able to select "vase" as a complement, just as "smash" and "shatter" can. Likewise, the child observes that the nouns "windshield" and "plate" behave similarly, as they can both be taken as complements by the three verbs "smash", "shatter" and "break", and that the noun "vase" appears similar to "windshield" and "plate" in so far as the three nouns can be taken as the complement by the two verbs "smash" and "shatter". This is a second line of observations that the child may use to support their conclusion that sentence (i) is a licit production. The strategy outlined above is a simplified illustration of how a Collaborative Filtering (CF) algorithm (reviewed in §2), which uses evidence of how related verbs and related nouns behave, can be used to infer whether a given verb can take a given noun as a complement.

The goal of the present study is to evaluate whether CF algorithms, a widely used method in artificial intelligence for developing recommender systems (Cacheda et al., 2011; Jalili et al., 2018), can be used to accurately model argument selection based on co-occurrence data obtained from a large English corpus (see §3 for details). The results of the experiments presented in this study (see §4) suggest that model-based CF algorithms perform well on the task of recommending which nouns a given verb can or cannot select as a complement, achieving AUROC of between 0.89 and 0.90 and surpassing a number of different baselines; notably, we found during model selection that the number of latent factors (and thus the size of the embedding vectors learned by these CF algorithms) was relatively small (at most 6 latent factors) as compared to the number of distinct verbs and nouns appearing in the analyzed co-occurrence data. Furthermore, we found that when we used k-means clustering to quantize the (per-verb and per-noun) embedding vectors learned by these CF algorithms, using the cluster of a particular verb or noun as a proxy for the verb or noun itself (as opposed to a distinct embedding vector per verb or per noun) yielded minimal loss of performance ($< 1\%$) even when we used a relatively small number of verb and noun clusters. (See §5 for details) Finally, our results suggest that model-based CF algorithms should be considered for use in modeling the inferences a language learner makes when considering problems of argument selection (see §6 for discussion).

2 A Review of Collaborative Filtering

Given a (finite) set of users, a (finite) set of items, and information about the ratings assigned by users to items (encoded in a *user-item rating matrix*), the task of a *recommender system* is to predict whether a given user would select a given item, or what rating the user would assign to the given item - these predictions can then be used to generate a list of recommended items for the given user (Bobadilla et al., 2013). *This study takes the users to be predicates (i.e. lexical verbs) and the items to be the arguments (i.e. common nouns) that a predicate may select as its complement (i.e. object).*

Content-based recommendation algorithms exploit similarities between the features associated with each item - e.g. a content-based recommendation algorithm would predict which arguments a given predicate can select by evaluating the semantic and syntactic features associated with the argument (Balabanović and Shoham, 1997; Lops et al., 2011). In the case that we do not have access to features associated with the items, we may employ Collaborative Filtering (CF) recommendation algorithms, which consider both similarities between the users and/or similarities between the items.[1] CF algorithms are traditionally divided into two groups, *memory-based* CF algorithms and *model-based* CF algorithms, which we will now describe.

Memory-based CF algorithms assume that similar users select similar items and assign similar ratings, with two users considered to be similar if they tend to assign similar ratings to items. Memory-based CF algorithms work by identifying a set of users who are similar to the target user – e.g. via the *k-Nearest-Neighbor* (KNN) algorithm, using the Pearson correlation coefficient as a similarity measure – and then predicting the rating the target user would assign to a given item by considering the ratings assigned by similar users, respectively weighted by the degree-of-similarity of each user to the target user (Schafer et al., 2007).[2]

Although Memory-based CF algorithms are widely used in practice, they face difficulty with respect to: (a) scaling as the sets of users and items grow, and (b) dealing with a sparse user-item interaction matrix (Adomavicius and Tuzhilin, 2005).

[1]See (Herlocker et al., 2004) and (Su and Khoshgoftaar, 2009) for surveys that review Collaborative Filtering algorithms; see (Burke, 2002, 2007) for a review of hybrid recommendation algorithms that combine aspects of content-based and CF algorithms.

[2]Memory-based CF algorithms can work to identify similar users, referred to as *user-based* collaborative filtering, or alternatively, work to identify similar items, which is referred to as *item-based* collaborative filtering (Sarwar et al., 2001).

Model-based CF algorithms, in which a predictive model is *learned*, address the aforementioned difficulties of Memory-based models. In particular, *latent factor models* such as Singular Value Decomposition (SVD) and Non-negative Matrix Factorization (NMF) employ dimensionality-reduction in which user and item profiles (i.e. the rows and columns of the *user-item rating matrix*) are embedded in a lower-dimensional space in which latent relationships between users and items become more explicit (Hofmann, 2004; Koren et al., 2009; Lü et al., 2012). In this way, latent factor models address two weaknesses of memory-based CF algorithms, scalability and sparsity, and for this reason the present study employs (latent factor) model-based CF algorithms.

3 Deriving a Verb-Noun Rating Matrix from Corpus Data

This study employs an English corpus, the *Corpus of Contemporary American English*. The COCA is a 385 million word corpus of (late 20th century and early 21st century) English derived from several domains including spoken language, fiction, magazines, newspapers, and academic articles (Davies, 2009, 2010).

The corpus was preprocessed as follows. First, we tokenized and segmented the corpus text into sentences. We then annotated the tokens in each sentence with Part-of-Speech (POS) tags.[3] Sentences without at least one verb and one common noun were then discarded. Finally we lemmatized the tokens in each sentence using the *TextBlob* python library. After preprocessing the COCA corpus, the text consisted of 75584272 words (not counting punctuation markers or numbers) segmented into 10087753 sentences, with 43429 distinct verb lemmas (derived from 60693 distinct lexical verbs) and 89541 distinct noun lemmas (derived from 105957 distinct common nouns).

Next, we derived a *verb-noun co-occurrence matrix* from the processed corpus as follows. The set of distinct (lexical) verb lemmas and the set of distinct (common) noun lemmas were indexed (using lexicographic ordering) as $\{v_1, v_2, v_3, ..., v_s\}$ and $\{n_1, n_2, ..., n_t\}$ respectively. Then the entry at row i and column j in the *verb-noun co-occurrence matrix* has value equal to the number of sentences in the (processed) corpus in which the (lemmatized) verb v_i and the (lemmatized) noun n_j co-occur, subject to the following constraints:

1. no other noun or verb appears between v_i and n_j;
2. the verb v_i must appear in the WordNet[4] verb database (Fellbaum, 1998);
3. the noun n_i must appear in the WordNet noun database (Miller, 1990, 1998);
4. for a sentence to be counted: (a) the verb v_i and the noun n_i must be the last verb and the last noun (respectively) in the sentence, with the verb preceding the noun; (b) neither a pronoun, a punctuation marker (e.g. comma, quotation mark, parenthesis) nor any of the tokens $\{who, what, that, by\}$ may appear between the verb and the noun.

To compensate for noise in the data, we required that a (verb, noun) pair must appear at least twice to be considered; thus, any entry in the co-occurrence matrix that has a value less than 2 was set to 0. Given our goal of predicting novel (verb, noun) pairings (i.e. where the verb serving as predicate may select the noun as an argument in complement position), we restricted our study to verbs and nouns for which there was evidence (in the corpus data) of co-occurrence with different nouns and verbs respectively. To this end, we removed rows corresponding to verbs that do not co-occur with at least two distinct nouns, and we removed rows corresponding to nouns that do not co-occur with at least two distinct verbs. After this process of reducing the verb-noun co-occurrence matrix, there remained 4172 rows, each corresponding to a distinct predicate lemma, and 12601 columns, each corresponding to a distinct argument column; the verb-noun rating matrix has a total of 716861 non-zero entries. In this way, we computed the *verb-noun co-occurrence matrix* from the (processed) COCA corpora.[5]

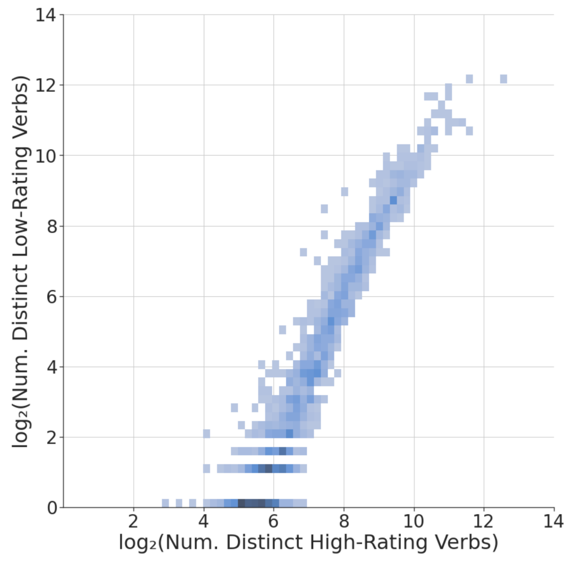
Finally, we constructed the *verb-noun rating matrix*, which serves as the input to the model-based CF algorithms.[6] We derived a distribution of verb occurrences and a distribution of noun occurrences, and from these two distributions we computed a joint distribution. (See Figure 1) We then used this joint distribution to compute the expected

[3]We used the POS-tags employed in annotating the PennTreebank.

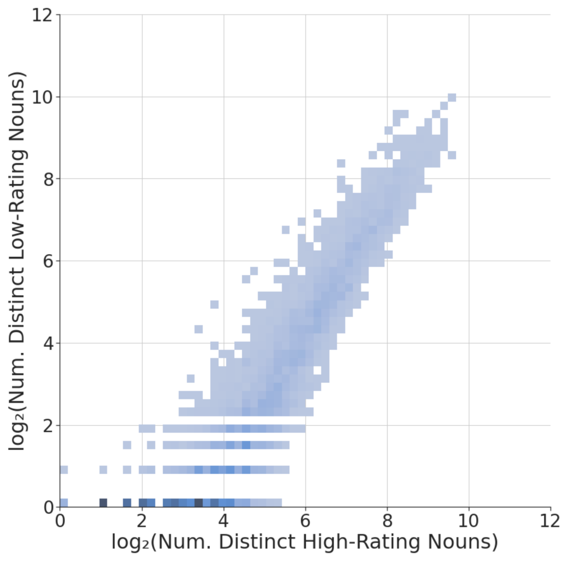
[4]See (Miller et al., 1990; Miller, 1995; Miller and Fellbaum, 2007) for reference on the WordNet database.

[5]The *verb-noun co-occurrence matrix* is stored as a list of three-tuples of (verb, noun, counts) for all (verb, noun) pairs for which counts is non-zero.

[6]This matrix corresponds to the *user-item rating matrix* in the terminology of collaborative filtering algorithms.



(a)



(b)

Figure 1: Distribution of ratings for verb-noun co-occurrences derived from the Corpus of Contemporary American-English. (1a) shows the distribution of the number of distinct nouns that are (high rating) or are not (low rating) expected to be selected by each distinct verb appearing in the corpus. Likewise, (1b) shows the distribution of the number of distinct verbs that are (high rating) or are not (low rating) expected to select each distinct noun appearing in the corpus.

number of counts for each (verb, noun) pairing. We assigned a *rating* to each (verb, noun) pairing for which there were a non-zero entry in the co-occurrence matrix. Whether a (predicate, argument) pairing was rated high or low corresponded to whether that predicate co-occurred with that argument more or less often than would be expected by chance. The ratings were thus assigned as follows: (i) a high rating, which has numerical value 2, was assigned if the value in the co-occurrence matrix was greater than the expected number of counts; (ii) a low rating, which has numerical value 1, was assigned otherwise.[7]

4 Experiment

The experiment detailed in this section addresses the question of whether model-based CF algorithms can be used to accurately predict which arguments (i.e. common nouns) a predicate (i.e. a lexical verb) may select.[8] We evaluated two different latent factor models, SVD and NMF.[9]

[7]The *verb-noun rating matrix* is stored as a list of three-tuples of (verb, noun, rating).

[8]Note that throughout this study, in accordance with our focus on understanding whether a *lexical verb* serving as a predicate may select a particular *common noun* as its (complement) argument, we will refer to (lexical) verbs as predicates and (common) nouns as arguments.

[9]We used the implementations of these models provided by the *Surprise* python library (Hug, 2020).

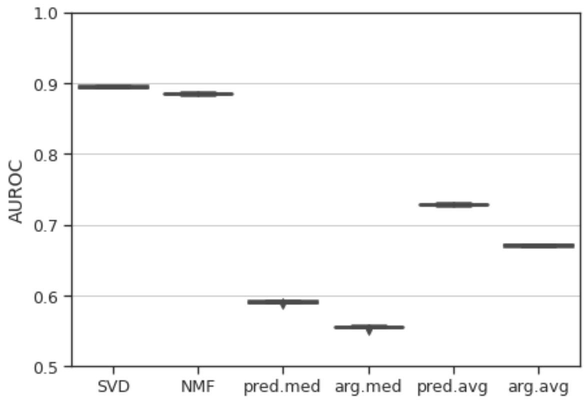


Figure 2: Performance of CF models and baselines.

4.1 Methodology

To train a model-based CF algorithm, we employed nested k-fold cross-validation with shuffling, with the outer loop used to evaluate trained models, and the inner loop used for model selection (hyperparameter tuning) and model fitting (i.e. training). The outer loop consists of a 5-fold cross-validation loop, with 20% of the data (i.e. entries in the *verb-noun rating matrix*) held out as a test data set, and the remaining data used for training and validation. The inner loop consist of a 5-fold cross-validation loop, with 20% of the data held out as a validation set, and the remaining data used for training. Model selection for both SVD and NMF consisted of opti-

Predicate	Recommended	Not Recommended
begin	dynasty, impeachment, simulation	mouth, side, weapon
challenge	doctrine, misconception, paradigm	mother, eye, place
destroy	battleship, habitat, rival	lot, week, word
elect	bishop, spokesperson, successor	life, place, world
perform	masterpiece, pushup, somersault	air, mother, town
sell	denim, postage, sushi	circumstance, editor, father
sit	cafeteria, parliament, veranda	employee, history, justice
spread	fluid, manure, paperwork	break, friend, point

Table 1: Examples of arguments that the SVD model with median performance (as measured by AUROC) does or does not recommend for selection as a complement for the listed predicate.

mizing the hyperparameter for the number of latent factors, $n_f \in [4, 21]$.[10] Models were evaluated for selection by computing the mean average error (MAE), a commonly used metric used for evaluating model-based CF algorithms. The output of a trained model thus consists of: (i) a mapping between predicates and embedding vectors of length $n_f + 1$; (ii) a mapping between arguments and embedding vectors of length $n_f + 1$; (iii) a matrix with $n_f + 1$ and $n_f + 1$ columns.

4.2 Results

We computed four baselines that, given a (predicate, argument) pairing (p, a) in the test set, make the following predictions:

- The *pred.med* and *pred.avg* baselines predict the median and mean values (respectively) of entries with predicate p (in the training set).
- The *arg.med* and *arg.avg* baselines predicts the median and mean values (respectively) of entries with argument a (in the training set).

The two trained collaborative filtering models and the four baselines each produce, for a given predicate and argument, a continuous value that we interpret as being a high or low rating based on whether is above or particular threshold value or not. We thus evaluated the performance of each model by computing the Area Under the Receiver Operating Characteristic curve (AUROC), which is presented in Figure 2.[11] Notably, the two CF algorithms, SVD and NMF, achieved an AUROC of 0.90 and 0.89 respectively, and both models outperformed each of the baselines - this suggests that both of these models perform well on the task of predicting which arguments a predicate can take as its complement. See Table 1 for examples of model predictions.

5 Analysis

Having seen that the two model-based CF algorithms performed well on the task of predicting which arguments a predicate is likely to select as its complement, we now turn to considering whether these CF models encode some knowledge of lexical-semantics. Lexical verbs may be said to cluster together into classes, with lexical verbs in the same class sharing similar syntactic and semantic properties (see the verb-classification established in (Levin, 1993)). The model-based CF algorithms that were evaluated in §4 produced distinct embedding vectors for each argument and for each predicate - this section analyzes whether these embedding vectors for verbs and nouns can be clustered into disjoint groups (of embedding vectors for verbs and nouns respectively). We will restrict our attention to the embedding vectors learned by the SVD and NMF models with median performance (with respect to AUROC).

For each of the two trained CF models, we used the *k-means* algorithm (Lloyd, 1982; Forgey, 1965; Jain, 2010) to group the embedding vectors for predicates and arguments into disjoint clusters of predicates and arguments respectively. The *k-means* algorithm was parameterized by the number of clusters the input (embedding) vectors should be grouped into. Let k_p denote the number of clusters the predicate embedding vectors are grouped into, and let k_a denote the number of clusters the argument embedding vectors are grouped into. A grouping of the predicates and

[10]Both models were trained over 350 epochs. For SVD we used a learning rate of 0.005 and a regularization rate of 0.02; for NMF we used a regularization rate of 0.06.

[11]See (Fawcett, 2006) for discussion on interpreting the Receiver Operating Characteristic curve.

arguments using particular values of k_p and k_a respectively is referred to as a *model-grouping*. We ran *k-means* clustering to compute model-groupings using $k_p \in \{10, 15, 20, \ldots, 55\}$ and $k_a \in \{10, 15, 20, \ldots, 100\}$, with clustering run five times for each selection of k_p and k_a.

We next evaluated how well a model-grouping could be used to predict whether a given predicate (lexical verb) selects a particular argument (common noun). Given a particular model-grouping with k_p predicate clusters and k_a argument clusters, for a verb-cluster, α, and a noun-cluster, β, let the $c_{\alpha,\beta}$ be the total number of co-occurrence counts for (verb, noun) pairs in (α, b) that have a high rating, and let $d_{\alpha,\beta}$ be the total number of co-occurrence counts for (verb, noun) pairs in (α, b); next, define the average rating $a_{\alpha,\beta}$ to be $\frac{c_{\alpha,\beta}}{d_{\alpha,\beta}}$, and let us suppose that all (verb, noun) pairs in (α, β) are assigned the rating $a_{\alpha,\beta}$. We can then recompute the AUROC for this model-grouping on the test data to evaluate the discriminatory power of the model-grouping; we refer to this metric as the *clustered-model AUROC*.

We identified model-groupings that achieved near-optimal clustered-model AUROC while using as few predicate clusters and argument clusters as possible. We did this as follows. Let the optimal clustered-model AUROC be the maximal clustered-model AUROC achieved by any model-grouping, and let a model-grouping be called *sub-optimal* if its clustered-model AUROC is less than 99% of the optimal clustered-model AUROC. Then a model-grouping is called *near-optimal* if (i) its clustered-model AUROC is within 1% of the optimal clustered-model AUROC, and (ii) decreasing either k_p or k_a yields a sub-optimal model-grouping. We computed the clustered-model AUROC for each model-grouping and identified model-groupings that were near-optimal. We found that near optimal performance (i.e. $< 1\%$ loss of maximal *clustered-model AUROC*) could be achieved using a relatively small number of clusters for verbs and nouns (relative to the dimensions of the *verb-noun rating matrix*). See Table 2 for statistics of both near-optimal model-groupings as well as model-groupings that obtained the optimal *clustered-model AUROC*.[12]

We also computed a summary statistic, the weighted average entropy of a clustering, that serves as a measure of how well (on average) the various verb and noun clusters in a model-grouping are able to cluster together similarly behaving verbs and nouns. This statistic was computed as follows. Given a particular model-grouping with k_p verb clusters and k_a noun clusters, for a verb-cluster, α, and a noun-cluster, β, the entropy for this pair of verb and noun clusters is:

$$e_{(\alpha,\beta)} = -\frac{c_{\alpha,\beta}}{d_{\alpha,\beta}} \log\left(\frac{c_{\alpha,\beta}}{d_{\alpha,\beta}}\right)$$

The entropy $e_{(\alpha,\beta)}$ measures how much information is needed (on average) to determine whether a randomly selected (verb, noun) pair in (α, β) has a high or low rating; note that since there are only two possible ratings (high and low), $0 \leq e_{(\alpha,\beta)} \leq 1$, and we expect a good (verb, noun) cluster-pair to have an entropy value closer to 0 than to 1, as that signifies that the members of the cluster pair are either majority high-rated or low-rated (verb, noun) pairs, which in turn suggests that these (verb, noun) pairs behave in a similar manner and are thus appropriate to cluster together. We then compute the weighted average entropy (also bounded between 0 and 1) of the model-grouping (k_p, k_a) is computed as:

$$W_{(k_a,k_b)} = \frac{\sum_{\alpha,\beta} c_\beta e_{(\alpha,\beta)}}{\sum_{\alpha,\beta} d_{\alpha,\beta}}$$

See Table 2 for the weighted average entropy for

model	k_p	k_a	AUROC	$W_{(k_a,k_b)}$	NMI
SVD	55	20	0.847	0.407	0.201
SVD	40	25	0.847	0.407	0.180
SVD	35	30	0.846	0.409	0.169
SVD	30	50	0.846	0.405	0.153
SVD	25	100	0.847	0.405	0.143
NMF	55	30	0.860	0.388	0.228
NMF	40	35	0.860	0.391	0.208
NMF	35	40	0.860	0.389	0.197
NMF	30	50	0.859	0.391	0.184
NMF	25	100	0.862	0.385	0.175
SVD	**55**	**95**	**0.854**	**0.383**	**0.198**
NMF	**55**	**100**	**0.868**	**0.372**	**0.227**

Table 2: Statistics for model groupings that are near-optimal with respect to AUROC on test data. Here k_p is the number of predicate (verb) clusters, k_a is the number of argument (noun) clusters. The two bolded rows at the bottom are the model groupings that had the maximal AUROC on the test data.

[12] See Figure 4 and Figure 5 in the appendix for the distribution of the *clustered-model AUROC* for each of the model-groupings, for the (median-performing) SVD and NMF models (respectively).

select model-groupings.[13]

Finally we sought to understand whether the information encoded within the verb embedding-vectors learned by a latent factor model aligns with traditional classification of verbs by their syntactic and semantic properties. To this end, we computed, for both the SVD and NMF models, the Normalized Mutual Information (NMI) between the clusterings of the predicate embedding vectors and the Levin verb classes.[14] The Levin verb classes were restricted to verbs that appear as verbs in WordNet, and (following (Li and Brew, 2008)) we removed verbs that appeared in more than two verb classes.[15] We also compared the NMI between the Levin verb classes against two baselines:

- Baseline A: group the predicates into k_p clusters of even size, with predicates randomly assigned to the clusters.
- Baseline B: create k_p clusters and randomly assign each predicates to one of the clusters.

We found the NMF model consistently had a higher NMI score than the SVD model (and thus aligned more closely with the Levin verb classification), and that both the SVD and the NMF model had higher NMI scores than the two baselines (see Figure 3). Note that the Levin verb classes were derived by considering the set of predicate-argument frames that a verb appears in, of which the argument-selection considered in this study (i.e. how a predicate selects an argument that serves as its complement) is a subset; thus, we expected at most a partial alignment between the predicate embedding vectors produced by the model-based CF algorithms and the Levin verb classes.

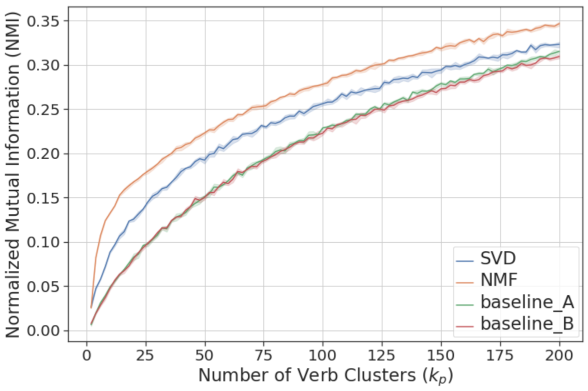


Figure 3: Normalized Mutual Information between Verb Clusters and Levin Verb Classes.

6 Conclusion

The results of this study suggest that model-based CF algorithms perform well on the task of inferring which (common) nouns a given (lexical) verb serving as a predicate can select as a complement. The two model-based CF algorithms that were evaluated on this task, SVD and NMF, achieved AUROC of 0.90 and 0.89 (respectively), indicating that they have good discriminatory power and surpassing the performance of several baselines. Notably, these two models achieved this level of performance using at most six latent factors, yielding embedding vectors of relatively low dimensionality as compared to the dimensionality of the *verb-noun rating matrix* from which they were derived. We also found that the embedding vectors yielded by the SVD and NMF models could each be clustered into a small number of disjoint groups with only a minuscule loss of performance (as measured by AUROC). Finally, we observed modest alignment of the verb clusters with the Levin verb classes as compared to several baselines. We believe that the results presented in this study warrant evaluation of whether model-based CF algorithms are suitable for modeling constituent selection, thereby going beyond inferring the arguments a predicate may select as its complement; such a set of CF models for constituent selection could be used to constrain the productivity of the rules that make up a constituent grammar, thereby yielding a system for learning a grammar from noisy corpus data.

Acknowledgments

Three anonymous reviewers are thanked for critically reading the manuscript and providing helpful comments.

References

Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749.

Marko Balabanović and Yoav Shoham. 1997. Fab:

[13]See Figure 6 and Figure 7 in the appendix for the distribution of weighted average entropy (for each model-grouping), for the SVD and NMF models (respectively).

[14]See (Strehl and Ghosh, 2002; Vinh et al., 2010) for a review of NMI. Note that NMI varies between 0, for no-alignment between two clusterings, and 1 for complete alignment between two clusterings.

[15]Consequently, there were a total of 48 verb classes, the largest having 186 verbs, the smallest having 1 verb, and the median-sized class having 16 verbs.

content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72.

Robert C Berwick, Paul Pietroski, Beracah Yankama, and Noam Chomsky. 2011. Poverty of the stimulus revisited. *Cognitive Science*, 35(7):1207–1242.

Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender systems survey. *Knowledge-based systems*, 46:109–132.

Robin Burke. 2002. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370.

Robin Burke. 2007. Hybrid web recommender systems. *The adaptive web*, pages 377–408.

Fidel Cacheda, Víctor Carneiro, Diego Fernández, and Vreixo Formoso. 2011. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web (TWEB)*, 5(1):1–33.

Mark Davies. 2009. The 385+ million word corpus of contemporary american english (1990–2008+): Design, architecture, and linguistic insights. *International journal of corpus linguistics*, 14(2):159–190.

Mark Davies. 2010. The corpus of contemporary american english as the first reliable monitor corpus of english. *Literary and linguistic computing*, 25(4):447–464.

Tom Fawcett. 2006. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

Christiane Fellbaum. 1998. A semantic network of english verbs. *WordNet: An electronic lexical database*, 3:153–178.

Edward Forgey. 1965. Cluster analysis of multivariate data: Efficiency vs. interpretability of classification. *Biometrics*, 21(3):768–769.

Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53.

Thomas Hofmann. 2004. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):89–115.

Nicolas Hug. 2020. Surprise: A python library for recommender systems. *Journal of Open Source Software*, 5(52):2174.

Anil K Jain. 2010. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666.

Mahdi Jalili, Sajad Ahmadian, Maliheh Izadi, Parham Moradi, and Mostafa Salehi. 2018. Evaluating collaborative filtering recommender algorithms: a survey. *IEEE access*, 6:74003–74024.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Howard Lasnik and Jeffrey Lidz. 2017. The argument from the poverty of the stimulus. *Oxford Handbook of Universal Grammar*, pages 221–248.

Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.

Jianguo Li and Chris Brew. 2008. Which are the best features for automatic verb classification. In *Proceedings of acl-08: hlt*, pages 434–442.

Stuart Lloyd. 1982. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. 2011. Content-based recommender systems: State of the art and trends. *Recommender systems handbook*, pages 73–105.

Linyuan Lü, Matúš Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang, and Tao Zhou. 2012. Recommender systems. *Physics Reports*, 519(1):1–49. Recommender Systems.

George A Miller. 1990. Nouns in wordnet: a lexical inheritance system. *International journal of Lexicography*, 3(4):245–264.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

George A Miller. 1998. Nouns in wordnet. *WordNet: An electronic lexical database*, pages 23–46.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244.

George A Miller and Christiane Fellbaum. 2007. Wordnet then and now. *Language Resources and Evaluation*, 41(2):209–214.

Massimo Piattelli-Palmarini and Robert C Berwick. 2012. *Rich languages from poor inputs*. OUP Oxford.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295.

J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617.

Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854.

Appendix

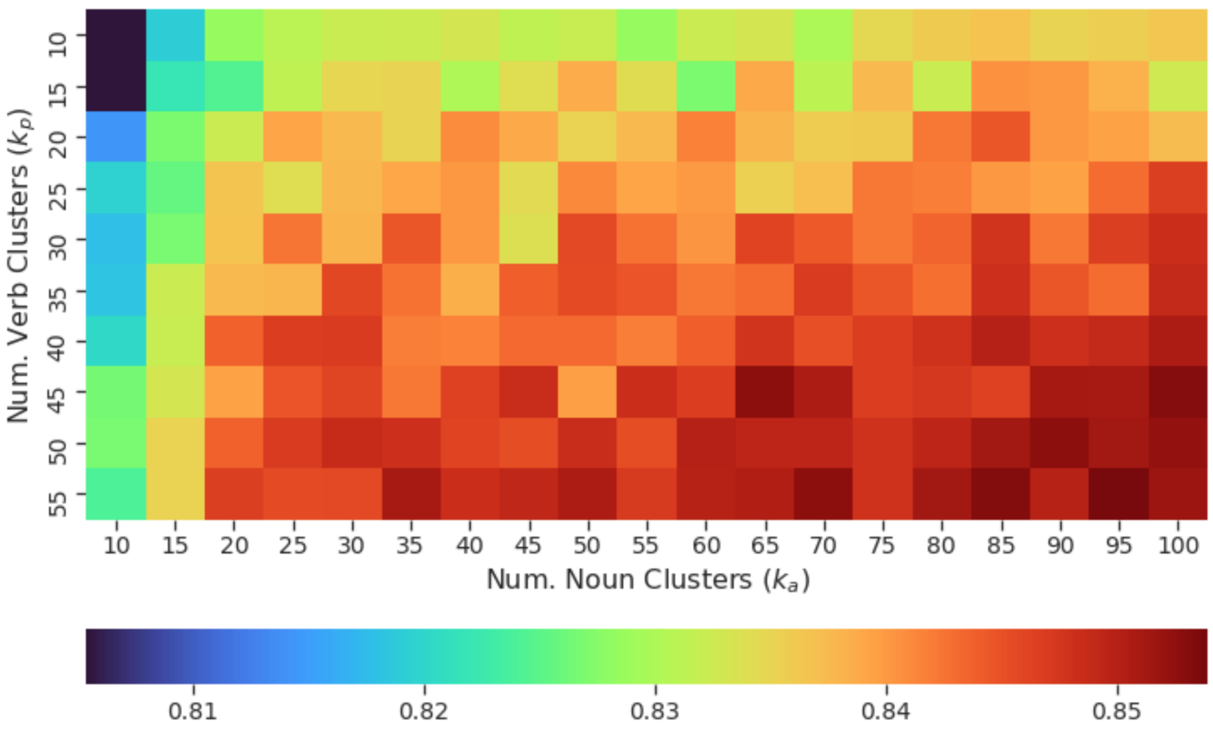


Figure 4: AUROC for each Model-Grouping derived from the median-performing SVD model.

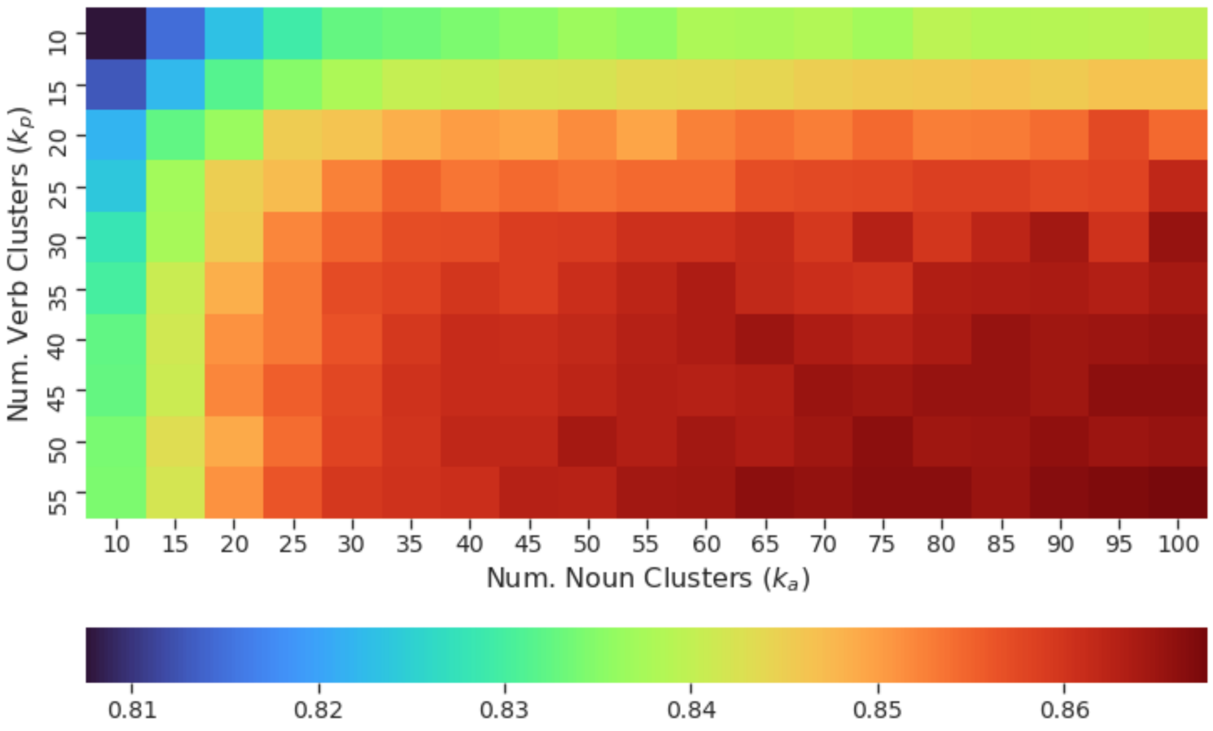


Figure 5: AUROC for each Model-Grouping derived from the median-performing NMF model.

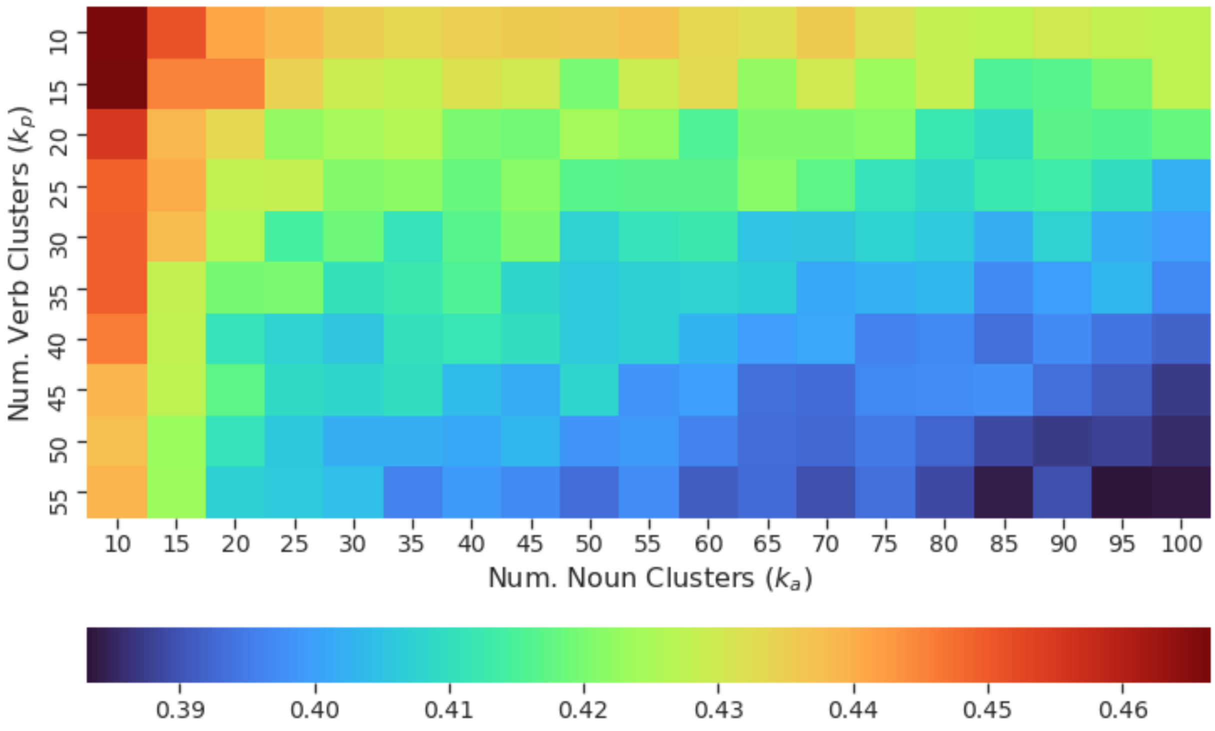


Figure 6: Weighted Average Entropy for each Model-Grouping of the median-performing SVD model.

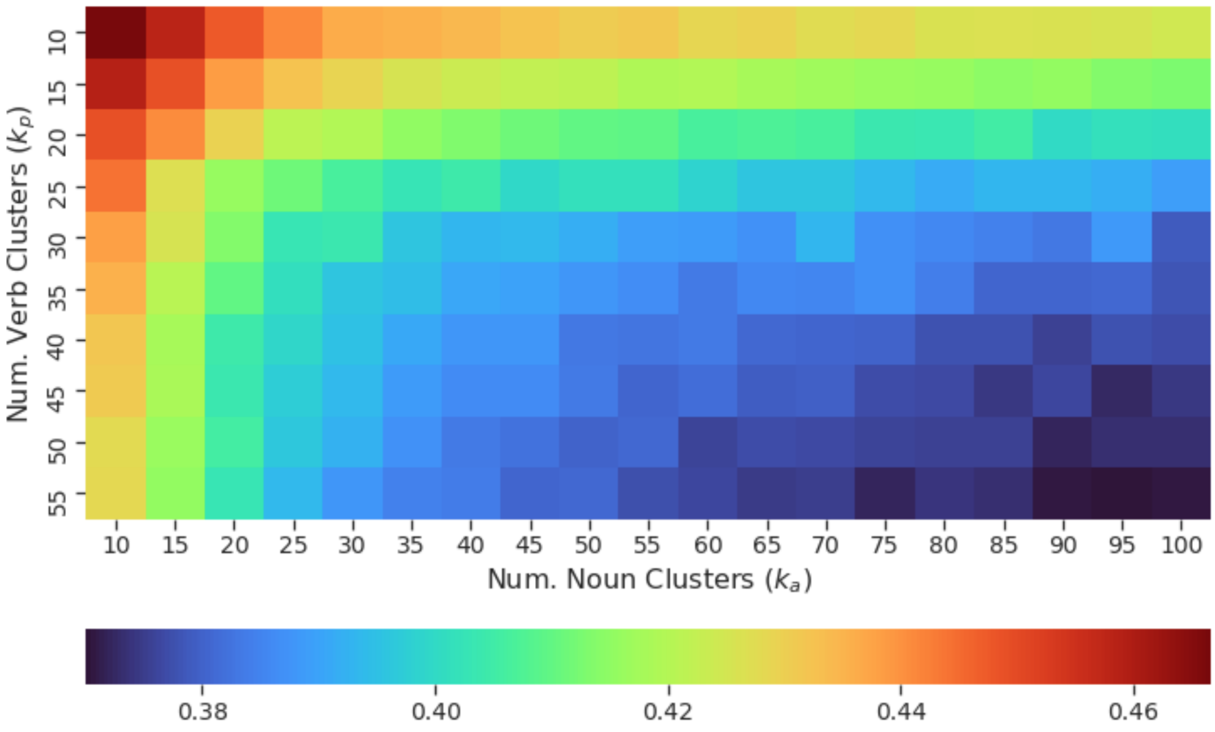


Figure 7: Weighted Average Entropy for each Model-Grouping of the median-performing NMF model.