Team 5 Final Report - MovieEdge

Rocko Graziano
rgraziano3@gatech.edu
Georgia Tech OMSCS

Daniel Klass
dklass3@gatech.edu
Georgia Tech OMSCS

Yi Sun
ysun428@gatech.edu
Georgia Tech OMSCS

Jonathan Tay
jtay6@gatech.edu
Georgia Tech OMSCS

ABSTRACT

We present **MovieEdge**, a recommendation system which provides an interactive visualization interface for users.

1 INTRODUCTION

Recommender Systems (RS) are ubiquitous in modern digital life. A fruitful area of RS research has been movie ratings, where a database of ratings is used to suggest new movies to watch. Most RS are based on Collaborative Filtering (CF), where user behavior and feedback is used to learn a model that can be used to make new recommendations. Interest in movie CF models peaked in 2007 with The Netflix Prize.

A key issue with CF systems is that they tend to be uninterpretable "black boxes", driven by the aggregate behavior of thousands or millions of users. Interpretability has been found to improve user acceptance and help detect bias in other settings [16]. Recently, researchers introduced MovieExplorer [25], which takes the first step toward interpretability by giving users the means to navigate a latent high (30) dimensional space (aka "taste space") derived from user preference data to find movie recommendations.

Innovations

We built a system, MovieEdge, which extends on MovieExplorer in two key ways:

We present recommendations based on a more accurate RS. Where MovieExplorer was driven by Matrix Factorization, we use a Word2vec model [15] which increases CF model performance [17] while leading to latent vector representations that are visually intuitive [15]. Word2vec is designed to generate embeddings, but not to provide predictions of user ratings. We apply Ridge Regression [20] to the embedding vectors generated from Word2vec to achieve state-of-the-art RS performance.

Additionally, while MovieExplorer allowed users to navigate the taste space, they did so without a visual reference. Two movies close in taste space can be said to be similar insofar as users who like one movie are likely to like the other. MovieEdge lets users visualize and interactively explore the taste space learned by the CF model. To our knowledge, this is the first attempt to visualize movie taste space. We overcome the following challenges to do this: 1) applying dimension reduction to plot data in 2D (using t-SNE); 2) keeping the visualization manageable despite a large number of movies (using clustering), and 3) designing a UI that allows users to navigate the taste space easily. We further augment exploration by providing supplemental data drawn from other sources.

2 RELATED WORK

Collaborative Filtering

Early CF systems were driven by nearest-neighbor similarity methods [5], [24], weighing similarity between user vectors of ratings. While neighborhood-based CF are easy to implement, they are neither accurate nor scalable.

Item-based CF methods are driven by item-to-item similarity. Amazon constructed a similar-items table by finding items that people tend to buy together [8], [24]. Scalability is achieved via offline preprocessing. However, item-based CF sacrifices user-centric interpretability for real-time performance .

The Netflix Prize introduced CF algorithms based on matrix factorization (MF) [3]. User-item ratings are viewed as a sparse matrix: rows represent users and columns movies. MF seeks to estimate ratings by approximating this matrix with a low-rank decomposition. MF is highly accurate for movie rating prediction and remains the predominant CF technique today [7].

Vector word embeddings (Word2vec) are widely adopted in Natural Language Processing. Two architectures, Continuous Bag-of-Words and Skim-gram models [14], [15], [21] were introduced to learn embeddings from words in sentences, capturing contextual similarity. This can be applied to user-item interaction data by treating items as words and the set of items rated highly by each user as a sentence [17].

Interactive Recommendations

Most RS lack interactivity. MetaLens [22] gives users session-specific control over the recommendation process, improving user satisfaction. MetaLens is limited to filtering or sorting the list of recommendations with user-specified criteria. The potential filters are hard-coded and express constraints rather than content preferences.

MovieExplorer [25], which incorporates user feedback in the RS, is the basis for MovieEdge. MovieExplorer allows the user to navigate the model's latent factor space by expressing their session-specific movie preferences. As the session progresses, MovieExplorer presents better recommendations. MovieExplorer's interactive exploration paradigm increased user satisfaction. However, MovieExplorer asks users to navigate the high dimensional latent factor space without a visual reference.

Visualization of Embedding Spaces

t-Stochastic Neighbor Embedding (t-SNE) projects high dimensional feature vectors into 2D visualizations [12]. t-SNE computes similarity scores between observations in the high dimensional feature space and finds a low dimensional embedding: points which are close in high dimensional space remain close in the projected space.

Embedding Projector [23] leverages t-SNE and PCA [18] to visualize high dimensional data. We will adopt this approach in the high dimensional taste space of our CF model, augmenting with supplemental data and user interactivity.

3 PROPOSED METHOD

MovieEdge was built iteratively over eight weeks, leveraging available machine learning packages and the Django web framework for Python.

Data Preprocessing

We downloaded 20 million MovieLens [4] ratings. Each rating comprises a user, a movie, the users' 5-star scale rating of that movie, and timestamp. We first binarize the rating data: each rating above a users' median rating is converted to +1; ratings below-median converted to 0. Binarized ratings are common in contemporary RS like Netflix [13]: they encourage users to rate more items, and they normalize the ratings for each users' average tendency. Additionally, binarizing ratings allow for efficient use of standard Word2vec implementations. To enhance the user experience, we utilized the Open Movie Database API [1] to download and present metadata about each movie, including the actors, director, year, and genre of the movie.

Recommender System

We implemented a novel CF system based on Word2vec. Word2vec is a state-of-the-art deep neural architecture designed to embed items in a high-dimensional space. Specifically, it is built on the idea that words

are represented by the context of words surrounding them. We trained the Word2vec model with `gensim` [19].

Our model views a users' rating history as a *sentence* where each movie is a *word*. Liked movies are one-hot-encoded and fed to the model as input. Due to time constraints, we did not attempt to develop a representation that distinguishes "disliked" movies from those that are merely unrated.

Words have order in sentences - "I walk to the store" makes more sense than "store the walk I to." We leverage the timestamp data to create ordered collections of rated movies. Ordering and subsequent windowing reduced training time and improved results by accounting for changing user tastes.

Once movies were embedded in latent taste space, the next step was to predict user movie preferences. The naive approach was to express a user as the average embedding vector of liked movies and use cosine similarity between a user's embedding vector and a movie to score each movie for each user. We call this approach "Word2vec cosine similarity" (W2V-cosine), and it produced mediocre results as noted in Section 4.

To improve our predictions, we designed a method called "Word2vec + Ridge Regression" (W2V+RR). In W2V+RR, we train a Ridge Regression [20] model on the embedding vectors generated by Word2vec, using the binarized rating label as the prediction target. A separate regression model is built for each user, representing the users' preferences. W2V+RR produces state-of-the-art performance but requires sufficient training data to fit the ridge regression. To address this, we fall back to W2V-cosine if the user has not submitted enough ratings. This hybrid approach allows us to avoid the "cold start" problem for new users when there are not enough ratings from the user to estimate the user model.

Visualize the model in 2D space

Our embeddings are in high dimensional space. We extracted the 64-dimension movie embeddings from the `gensim` model and applied dimensionality reduction via t-SNE to visualize our data in 2D space. To accelerate the t-SNE processing, typically an $O(N\ log(N))$ algorithm, we used the $O(N)$ FFT based algorithm described in [9].

Our dataset of 27,278 movies is challenging for users to navigate, necessitating clustering for ease of visualization. We chose to use Agglomerative clustering [26], which allowed us to smoothly merge and break up clusters as the user explores the taste space.

User Interface

All computation (embedding movies in taste space, dimension reduction, and hierarchical clustering) can be carried out in advance and saved for reuse, allowing us to minimize computation during the visualization.

Upon entering the site, a user is presented with an initial selection of ten movies and the associated taste space (see Figure 1).

The User Interface is divided into two panes. In the right pane, we visualize the movies as points in 2D space. The user navigates this space by panning and zooming. As the user zooms out, we automatically reduce clutter and improve rendering performance by displaying clusters rather than individual movies. As the user zooms in, these clusters break up, eventually into discrete movies. Hovering over a point in the visualization shows the movie's thumbnail and metadata (rating, director, and genre).

A user selects a movie by clicking its thumbnail on the left panel. This updates the user's location in taste space. Users are asked to Like or Dislike movies in the left pane, and this information is fed to our RS to iteratively improve the estimation of user tastes. A "Random Movies" button provides additional choices unrelated to current likes/dislikes. If they supply enough data, we switch from W2V-cosine to W2V+RR for more accurate recommendations.

Two strategies ensured a smooth rendering of the visualization. First, we precomputed cluster

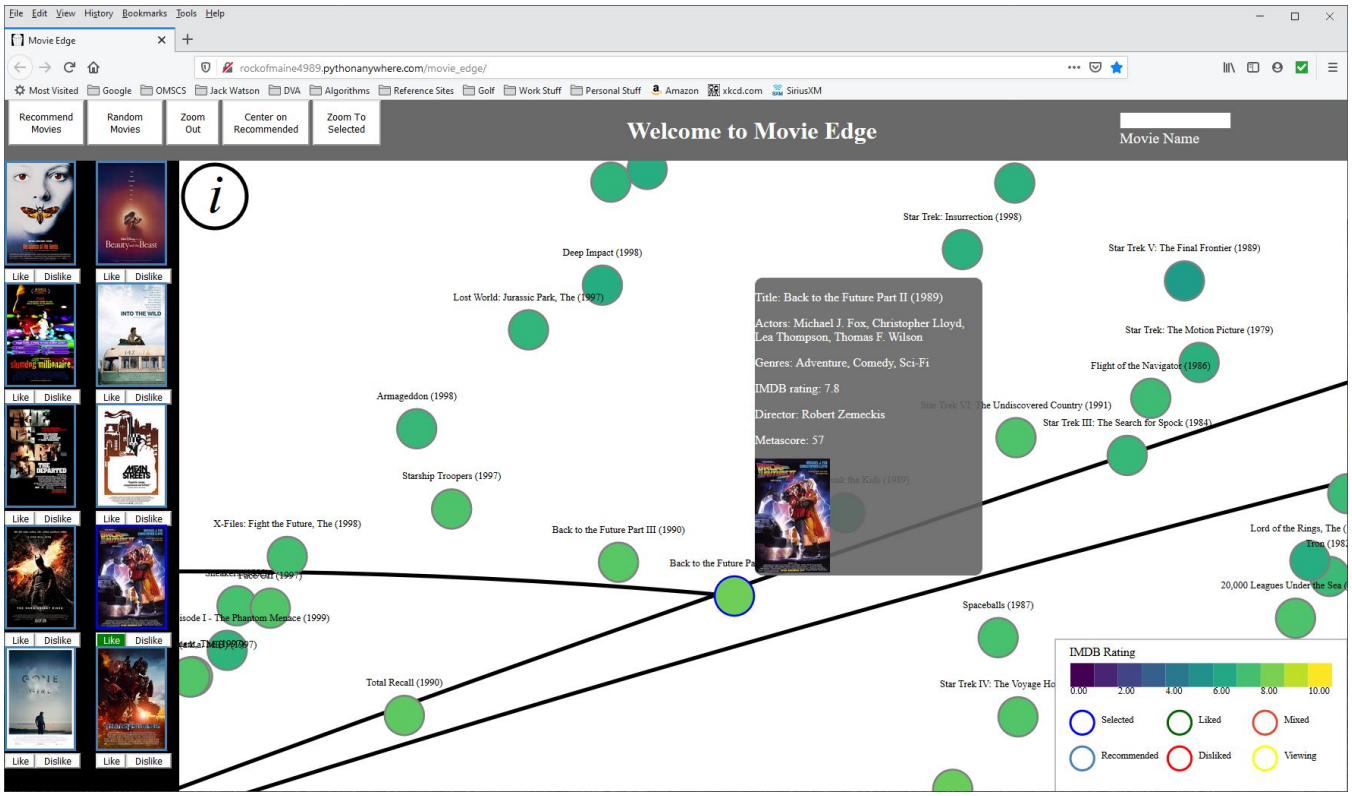


Figure 1: MovieEdge Main Screen

locations and plotted groups of movies as a single point, with size proportional to the number of movies in the cluster rather than as individual movies. Second, we deferred rendering updates as much as possible. For example, we found the browser unresponsive when too many SVG text elements (like labels) are rendered at the same time — adding labels only when the movies come into the viewport improved performance.

4 EXPERIMENTS AND RESULTS

The goals of MovieEdge are to improve on MovieExplorer by providing better recommendations and interactive visualization of the taste space. We evaluated the former by testing various models, including the CF model used in MovieExplorer, on an out of sample dataset, the latter by conducting a brief user survey.

Improving Recommendations

Matrix Factorization (MF) methods are common in CF recommendation models. MF expresses the (user, movie, rating) data as a sparse rating matrix and estimates a low-rank decomposition of this matrix, which can then be used to predict ratings for unseen (user, movie) pairs. Matrix Factorization approaches we tested include:

- Non-negative MF (NMF)
- Non-negative Matrix Tri-Factorization (NMTF)
- Singular Value Decomposition (SVD)
- SVD++

More comprehensive descriptions of these models can be found in Appendix A. MovieExplorer uses a 30-dimensional NMF model, so NMF is our performance baseline.

Our ratings data was split into 70% training, 15% validation, and 15% test sets, stratified by users, ensuring all users are represented in each dataset.

We trained candidate models on the training set and selected hyperparameters using the validation set. We did not use cross-validation given the size of the rating dataset.

Once hyperparameters were selected, we retrained the models with the combined training and validation sets (In-Sample) and computed the Receiver Operating Characteristic Area Under the Curve (AUC) on the test data (Out-of-Sample).

Method	In-Sample AUC	Out-of-Sample AUC
Baseline (mean ratings)	0.7644	0.7540
NMF	0.7381	0.7293
NMTF	0.6342	0.6300
SVD	0.8254	0.7822
SVD++	0.8308	0.7992
W2V-cosine	0.6755	0.6588
W2V+RR	**0.8801**	**0.7906**

Table 1: AUC Scores for Evaluated Models

SVD++ had the best AUC scores, followed closely by our novel W2V+RR model. However, W2V+RR offers three advantages over SVD++. First, each SVD++ model took 12-13 hours to train while W2V+RR took 1 hour. Second, SVD++ has significant cold start issues and cannot predict ratings for a new user without any reviews while W2V+RR can fall back to W2-cosine when necessary.

Finally, where SVD++ would have to be completely retrained upon the addition of a new movie or user, W2V+RR comes in two independent components (movie embedding and user preference ridge regression). These can be easily updated independently with new user-movie-rating data. In particular, as Ridge Regression on small datasets can be trained extremely fast, MovieEdge can train the Ridge Regression component of W2V+RR dynamically as the user interacts with the tool.

Appendix B (Page 10) presents the validation curves of our models.

User Survey

We conducted a survey to gauge user reaction to MovieEdge. After a brief introduction and video, 18 participants were asked to respond to seven quantitative and two open-ended questions. Two versions of MovieEdge were presented: one with the visualization graphic, the second limited to just movie thumbnails. The results were mixed: 77.78% saw a need for MovieEdge, but only 18% were ready to recommend our prototype to others.

Question	With Visualization	Just Thumbnails
Preferred version	44.44%	55.56%
Positive Reaction	72.22%	77.78%
Innovative Product	88.89%	72.22%

Table 2: User Responses

The results, combined with the open-ended feedback, suggest that while people found our approach innovative, there is work to be done to make the UI more intuitive and easier to navigate. The complete survey results may be found in Appendix C (Page 11).

Other Observations

Movie Similarities. Figure 2 shows a visualization of Star Trek, zoomed to the finest level. Note that our embedding has identified seven Star Trek movies in this view: fans of one Star Trek movie are likely to rate the rest highly. Six of these are based on The Original Series. More recent Star Trek movies like “Star Trek: Into Darkness” are located further away in taste space, indicating that the reboot has a different fan base.

Year Cohort Effects. Figure 3 shows that MovieEdge often displays movies clustered by year of release. Recall that the visual position of these movies comes from a t-SNE projection of a 64-dimensional Word2vec embedding. In particular, movies that are rated positively together tend to have similar embedding vectors, and thus similar locations after t-SNE projection. We hypothesize that we

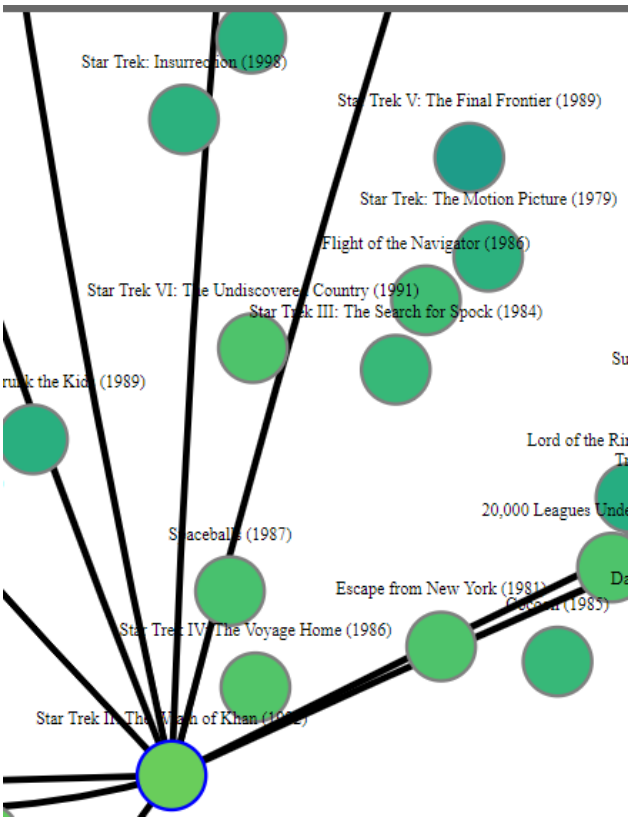


Figure 2: Star Trek in the taste space

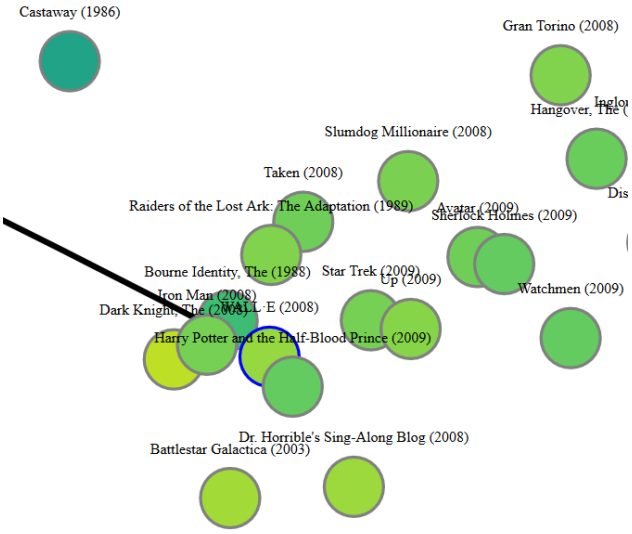


Figure 3: Movie Cohorts influenced by year

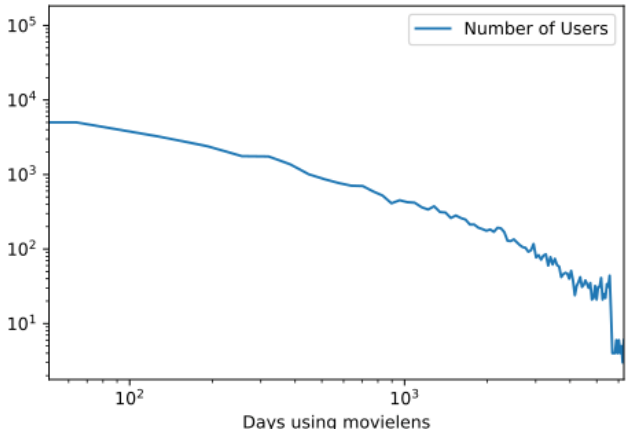


Figure 4: MovieLens - Days of Use

are seeing windowing effects from Word2vec: our Word2vec embedding is designed to predict the next liked movie given the last 32 liked movies, so the temporal order of movie release dates has a disproportional effect on the embedding.

Another phenomenon exacerbates the year-cohort clustering issue. Figure 4 is a log-log plot of the count of MovieLens users against the length of time they spend on the platform (the number of days between their first and last rating). We observe a characteristic statistically significant (p-value 3.87%) power-law profile with an exponent of -1.23. This shows that the vast majority of MovieLens raters do not use the platform for very long, suggesting the data is dominated by user samples where a set of recently seen movies are rated. This would drive the embedding vectors of movies with similar release dates together.

5 CONCLUSIONS AND DISCUSSION

We have successfully developed a novel CF Recommender System (W2V+RR) that achieves near state-of-the-art performance while offering faster training and easier updates. However, we noted a distinct "cohort" problem presented by using a sliding chronological window over user ratings: users seem to rate movies that were in the theater at a similar time together. This is something we didn't anticipate but intuitively makes sense.

Increasing window size, or perhaps randomizing movies within a window would alleviate this.

We were successful in using t-SNE dimensionality reduction and clustering to display similar movies near one-another. The ability to visually navigate a taste space is a powerful tool and allows for more insight than the typical movie-poster-based display.

The use of hierarchical clustering did impact user adoption. The meaning of a movie is obscured as clusters abstract away much of what makes a movie a piece of culture: script, plot, visuals. We try to convey a reasonable amount of this via tooltips showing common genres and actors/actresses but realize that conveying information at such an abstracted view is difficult.

Despite our efforts to make the process intuitive, the visualization of taste space was poorly received by some users, who found it unnecessary to the task of finding a movie recommendation. However, other participants appreciated the visual exploration aspect of MovieEdge after watching our demo video. We believe that MovieEdge is best suited for exploration-related tasks where the user is explicitly seeking novelty and serendipity.

6 TEAM MEMBER EFFORT

All team members contributed equally to this effort, collaborating through #slack and providing development and report content. Focus areas were:

- Daniel – Word2vec models, t-SNE, Django web framework, movie grid;
- Jonathan – Matrix Factorization experiments, generating graph of taste space;
- Rocko – User Interface, written deliverables (reports, poster), project management;
- Yi – Word2vec models, development of W2V-cosine and W2V+RR, live inference and navigation.

A functioning prototype may be found at http://rockofmaine4989.pythonanywhere.com/movie_edge

REFERENCES

[1] [n.d.]. The Open Movie Database. https://www.omdbapi.com/. (Accessed on 10/03/2019).

[2] Andrej Čopar, Blaž Zupan, and Marinka Zitnik. 2019. Fast optimization of non-negative matrix tri-factorization. PloS one 14, 6 (2019), e0217994. https://github.com/acopar/fast-nmtf

[3] Simon Funk. 2006. Netflix update: Try this at home. https://sifter.org/~simon/journal/20061211.html

[4] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis) 5, 4 (2016), 19. https://grouplens.org/datasets/movielens/

[5] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. 1999. An algorithmic framework for performing collaborative filtering. In 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999. Association for Computing Machinery, Inc, 230–237.

[6] Nicolas Hug. 2017. Surprise, a Python library for recommender systems. http://surpriselib.com.

[7] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 426–434.

[8] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. IEEE Internet computing 1 (2003), 76–80.

[9] George C. Linderman, Manas Rachh, Jeremy G. Hoskins, Stefan Steinerberger, and Yuval Kluger. 2019. Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data. Nature Methods 16, 3 (2019), 243–245. https://doi.org/10.1038/s41592-018-0308-4

[10] Bo Long, Zhongfei Mark Zhang, and Philip S Yu. 2005. Co-clustering by block value decomposition. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. ACM, 635–640.

[11] Xin Luo, Mengchu Zhou, Yunni Xia, and Qingsheng Zhu. 2014. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. IEEE Transactions on Industrial Informatics 10, 2 (2014), 1273–1284.

[12] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. Journal of machine learning research 9, Nov (2008), 2579–2605.

[13] Nathan Mcalone. 2017. Netflix has replaced its 5-star rating system with 'thumbs up, thumbs down' – here's why. (5 April 2017).

https://www.businessinsider.com.au/why-netflix-replaced-its-5-star-rating-system-2017-4?r=US&IR=T

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013).

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems. 3111–3119.

[16] Christoph Molnar. 2019. Interpretable Machine Learning - A Guide for Making Black Box Models Explainable. https://christophm.github.io/interpretable-ml-book/, Chapter Interpretability.

[17] Makbule Gulcin Ozsoy. 2016. From word embeddings to item recommendation. arXiv preprint arXiv:1601.01356 (2016).

[18] Karl Pearson. 1901. LIII. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2, 11 (1901), 559–572.

[19] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[20] Ryan Michael Rifkin. 2002. Everything old is new again: a fresh look at historical approaches in machine learning. Ph.D. Dissertation. MaSSachuSettS InStitute of Technology.

[21] Xin Rong. 2014. word2vec parameter learning explained. arXiv preprint arXiv:1411.2738 (2014).

[22] J Ben Schafer, Joseph A Konstan, and John Riedl. 2002. Meta-recommendation systems: user-controlled integration of diverse recommendations. In Proceedings of the eleventh international conference on Information and knowledge management. ACM, 43–51.

[23] Daniel Smilkov, Nikhil Thorat, Charles Nicholson, Emily Reif, Fernanda B. Viégas, and Martin Wattenberg. 2016. Embedding Projector: Interactive Visualization and Interpretation of Embeddings. arXiv:stat.ML/1611.05469

[24] Brent Smith and Greg Linden. 2017. Two decades of recommender systems at Amazon. com. Ieee internet computing 21, 3 (2017), 12–18.

[25] Taavi T Taijala, Martijn C Willemsen, and Joseph A Konstan. 2018. Movieexplorer: building an interactive exploration tool from ratings and latent taste spaces. In Proceedings of the 33rd Annual ACM Symposium on Applied Computing. ACM, ACM, 1383–1392.

[26] Joe H Ward Jr. 1963. Hierarchical grouping to optimize an objective function. Journal of the American statistical association 58, 301 (1963), 236–244.

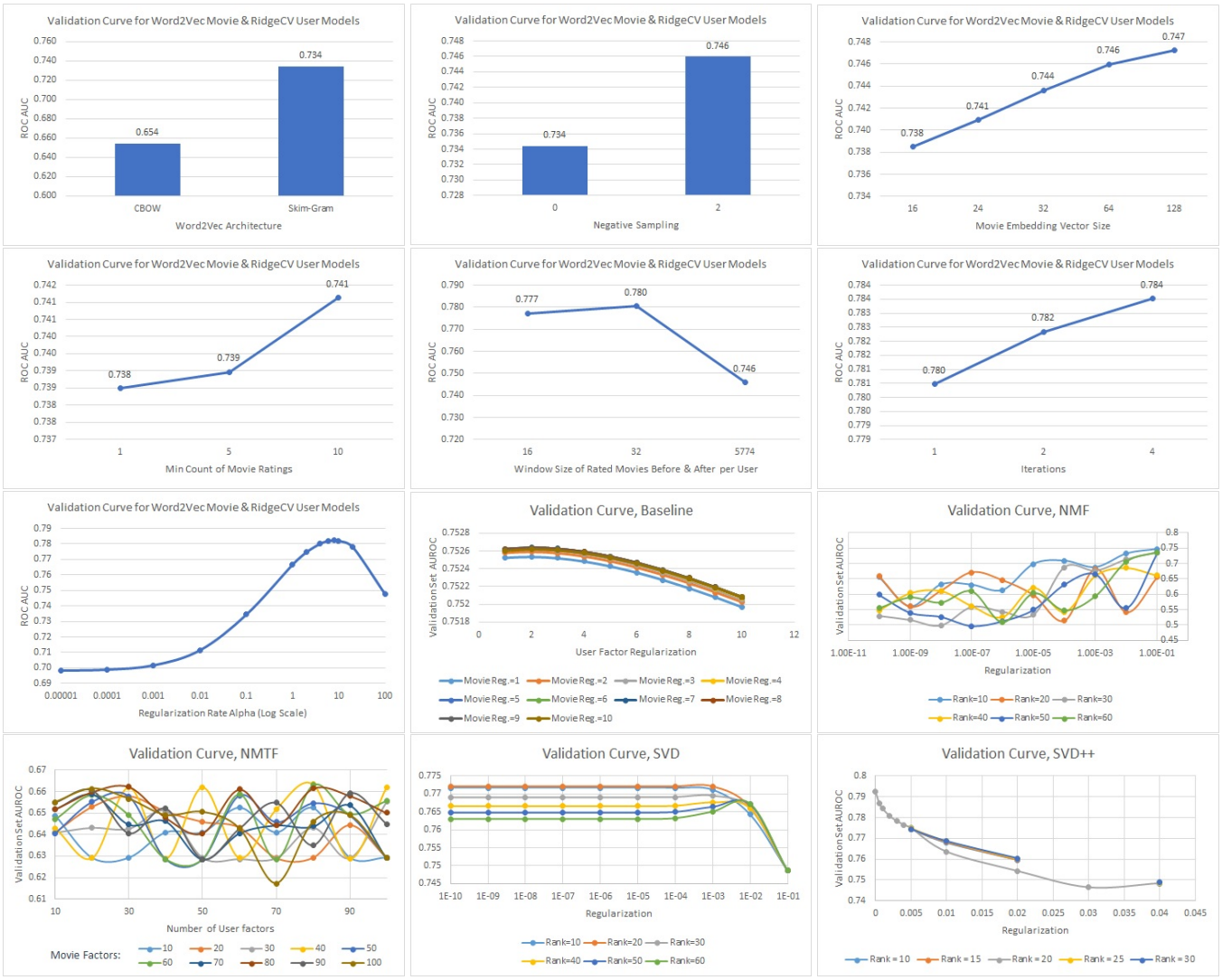
Appendices

A MATRIX FACTORIZATION

MF methods express the (user, movie, rating) data as a sparse rating matrix and attempt to estimate a low-rank decomposition of this matrix, which can then be used to predict ratings for unseen (user, movie) pairs. We evaluated five methods. Unless otherwise stated, these used implementations from the scikit-Surprise package ([6]).

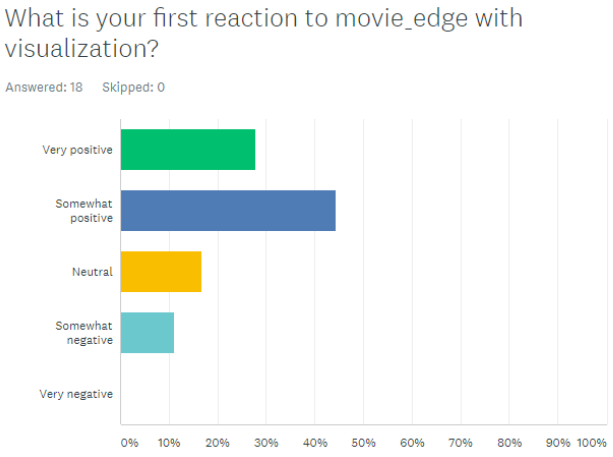
- Baseline - is not strictly speaking an MF method. Rather, baseline is a simple model that expresses a rating as the sum of a global mean rating, a user's mean rating, and the movie's mean rating with all three terms are jointly estimated. Baseline is an integral part of many MF based CF models ([6]), with the MF being estimated against deviations from baseline predictions. Baseline principally serves to de-mean each row and column of the rating matrix.
- Non-negative MF (NMF) - is a standard MF method, and is used as the base RS in MovieExplorer. In NMF, the rating matrix is decomposed into two low-rank terms, with the constraint that all elements of both components must be non-negative. This constraint has been shown to regularize the decomposition [11].
- Non-negative Matrix Tri-Factorization (NMTF) - introduced by [10], NMTF decomposes the rating matrix into three components: an assignment of users to clusters, an assignment of movies to clusters, and an affinity between each cluster of users and movies. NMTF has the advantage of good interpretability, but can be hard to estimate on large datasets. We leverage [2] to greatly speed up model estimation.
- Singular Value Decomposition (SVD) - is based on [3], an early MF approach to CF. In SVD, we decompose the rating matrix into user and movie factors, which are jointly estimated in the same latent space. A rating's deviation from the baseline prediction is then the dot product of the user and movie factors for that user/movie pair.
- SVD++ - is an advanced SVD method developed by the winning team of the Netflix prize competition [7]. SVD++ enhances SVD by using a more expressive user model. In this case, the user is expressed as a sum of a personal taste vector and a function of the movies the user has chosen to see (and rate).

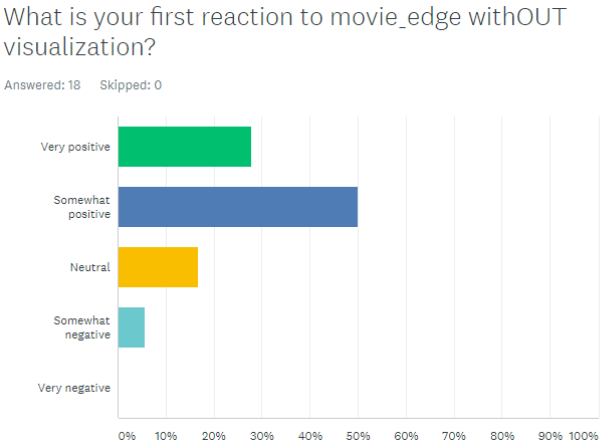
B VALIDATION CURVES

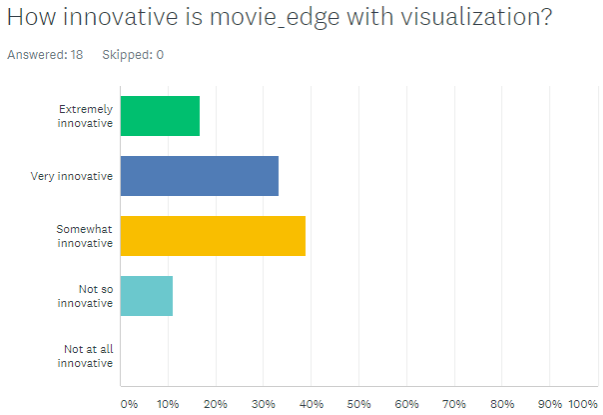


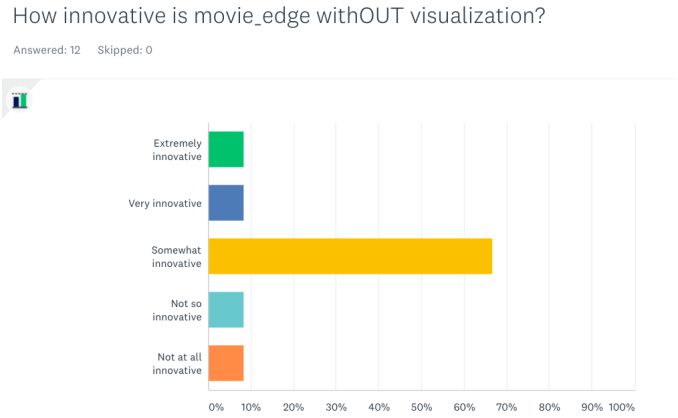
The validation curves represent the various combination of hyper-parameters evaluated for multiple models. **SVD++** proved to be the most accurate, while our novel **W2V+RR** model achieved comparable accuracy with significant advantages in training time and interactive usability.

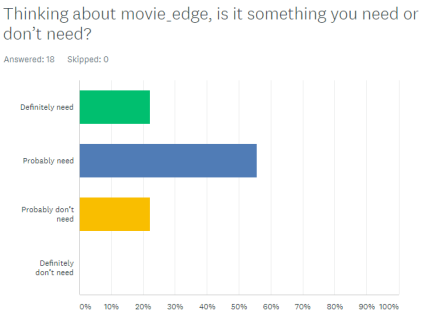
C USER SURVEY

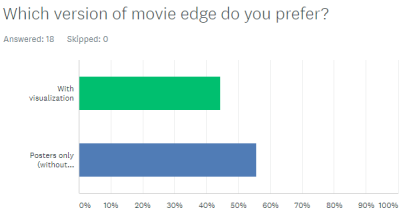


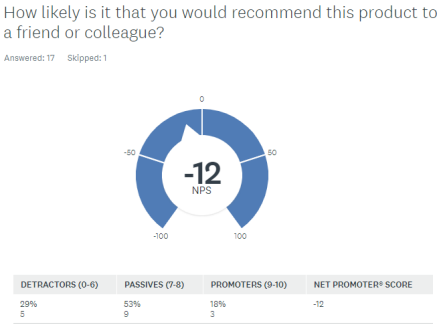












Survey participants (fellow classmates and a selection of family and colleagues) were introduced to MovieEdge, asked to watch a small demo video, and then test two versions. The first displayed movie thumbnails & captured ratings; the second included our interactive visualization of the movie "taste space."