DeepAniNet: Deep NLP-based Representations for a Generalizable Anime Recommender System

Stanford CS224N Custom Project

Michael Sun
MCS
Stanford University
msun415@stanford.edu

Tai Vu
Computer Science
Stanford University
taivu@stanford.edu

Robert Yang
Computer Science
Stanford University
bobyang9@stanford.edu

Abstract

Traditionally, recommendation systems require a long history of user-item interactions in the form of a large preference matrix to perform well, making them impractical without large datasets. We aim to build a successful content-driven recommendation system that takes a hybrid ground between collaborative filtering (CF) approaches based off of a preference matrix, and nearest neighbor approaches based off of self-supervised embeddings. Specifically, we develop a deep learning, NLP-based anime recommender system named **DeepAniNet** on top of representations of anime shows called **anime2vec**. We explicitly train our model to reconstruct user-anime relevance scores, for shows with few or zero interactions. Our goal is to demonstrate that deep NLP approaches can extract rich content features to improve both a recommender system's performance and ability to generalize to new users and anime.

We are mentored by Andrew Wang, with no external collaborators or courses that share this project.

1 Introduction

Large pretrained models have delivered breakthroughs in areas across NLP, from Question Answering to Machine Translation. They have enabled a new generation of consumer-facing applications. However, recommendation systems are still reliant on large datasets that are unable to be easily collected or found. With smaller datasets, the shortcomings of traditional recommendation systems begin to appear. Hence, we motivate the merits of a content-based approach with the problem of cold start, instances for which few or no interaction history is available (such as new users or added items). In particular, our approach, **DeepAniNet**, combines the DropoutNet architecture [10] with pretrained text embeddings to train content encodings and reconstruct relevances for users and items, thereby allowing the system to make recommendations solely on the basis of content without the need for separate user features. We demonstrate the system's generalization to unseen anime shows, superior metrics that beat the choice of content featurization in the original DropoutNet paper - TF-IDF, and desirable qualitative properties. In addition, it will be to our knowledge the first ever deep NLP driven anime recommendation engine.

2 Related Work

In recent decades, recommender systems have been an active area of machine learning research. A popular solution to building recommendation engines is model-based collaborative filtering. This method builds compact representations of user-item interactions in the training data, thus making suggestions for each individual based on other people's preferences. However, model-based collaborative filtering suffers from sparsity, and struggles with cold start entities: users or shows with on previous preference data.

There have been a number of other hybrid approaches to the problem of cold start in recommendation systems. Some of them are collaborative topic regression (CTR) [11], collaborative topic poisson factorization (CTPF) [5], and collaborative deep learning (CDL) [12]. However, they introduce highly complex objective functions to incorporate additional content and preference terms. In addition, these models only handle cold start items without paying attention to cold start users. Meanwhile, many research groups have applied deep learning to developing collaborative filtering-based recommendation engines, such as DeepMusic [9] and recurrent recommender networks [13]. Nevertheless, there have been limited attempts to address the cold start problem directly in the training procedure

We tackle this problem by following DropoutNet [10], which showed the feasibility of generalizing to cold start users and items by employing dropout at training time to reconstruct user-item relevance scores. Specifically, we extend their work with deep trainable content encodings, including BERT and graph embeddings, and demonstrate superior performance on the task of anime recommendation to static features.

3 Approach

Preprocessing

We first split our dataset of 10000 top-rated anime shows into $M^{train}, M^{val}, M^{test}$ with a $8:1:1$ ratio. We followed DropoutNet's paper by applying weighted matrix factorization (WMF) to approximate $R \in \mathbb{R}^{N \times M^{train}}$, the training preference matrix, by using $R_{u,v} \approx U_u^{train}(V_v^{train})^T$, where U^{train} and V^{train} are dense, low dimensional latent representations the N users and M^{train} train set items.

Loss Function

We defined $f_U, f_V, f_{\phi^U}, f_{\phi;V}$ as deep neural networks. The user and item latent vectors, U_u and V_v, are passed through f_U, f_V. The user and item feature vectors, ϕ_u^U, ϕ_v^V, are passed through f_{ϕ^U}, f_{ϕ^V} respectively. We then concatenated the latent and feature vector representations as $[f_U(U_u); f_{\phi^U}(\phi_u^U)]$ and $[f_V(V_v); f_{\phi^V}(\phi_v^V)]$ and pass them through f_U and f_V respectively to obtain $\hat{U}_u$ and $\hat{V}_v$. The loss was defined as

$$O = \sum_{u,v} (U_u^T V_v - \hat{U}_u^T \hat{V}_v)^2,$$

, which we optimized via stochastic gradient descent.

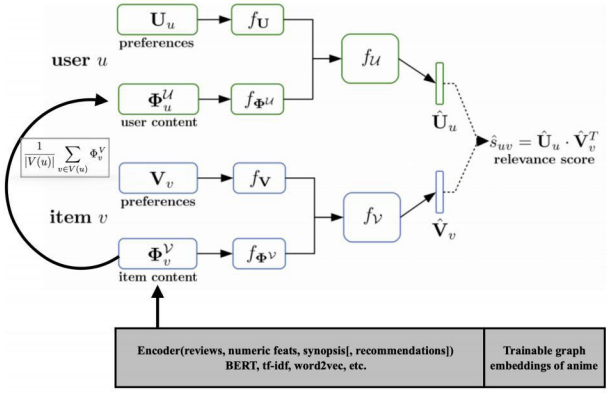

Figure 1: DeepAniNet: Item content featurized by concatenating encoding with graph embedding (The two thick arrows show our additions to the DropoutNet paper's diagram [10])

We employed our own design decision $\phi_u^U \approx \frac{1}{|V(u)|} \cdot \sum_{v \in V(u)} \phi_v^V$ for all training, validation, and test sets.

At training time, we explicitly trained the model's ability to generalize to new items by performing one of three possible options:

- Leave as is.
- Dropout: $(V_v, \phi_v^V) \to (0, \phi_v^V)$ (with prob_dropout).
- Transform: $(U_u, \phi_u^U) \to \left(\frac{1}{|V(u)|} \cdot \sum_{v \in V(u)} V_v, \phi_u^U\right)$ (with prob_transform).

We performed dropout to train for cold start (since $V_v^{val/test} = 0$) and transform to employ the trick introduced in the original paper of $U_u \approx \frac{1}{|V(u)|} \sum_{v \in V(u)} V_v$ for the val and test sets. We trained on this loss applying various choices encoder choices for ϕ^V, which we called **anime2vec**.

Model Architecture and Code

We implemented DropoutNet with item dropout and transform and ran experiments with both static and dynamic embeddings for ϕ^V. Dynamic embeddings include the choice of encoder and/or graph embedding. For the encoder, we fine-tuned BERT with various hyperparameters and training techniques. For the graph embeddings, we tried the following:

- GCNConv (with no edge features) by initializing each anime as either a one-hot vector or the same as ϕ^V.
- GINE by representing each edge as an encoding of all recommendations written for the anime pair and the individual anime initialization as above.

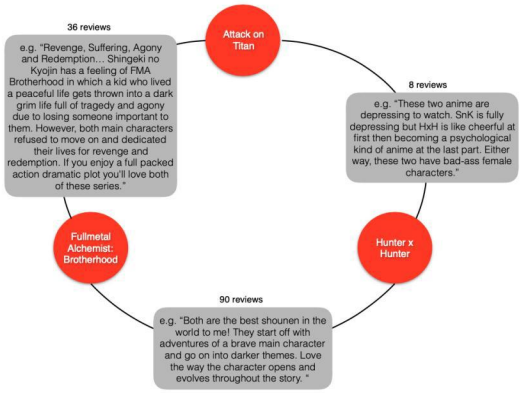

Figure 2: The Anime Graph: 26920 undirected edges for 10000 animes (very sparse)

We wrote our own code for the data discovery, collection, and processing pipeline here, web scraping here, the web app here, and model training here. Besides, we adapted and modified the code for pretraining BERT with masked language modeling from the HuggingFace documentation [7]. Additionally, our web app demo can be found here.

4 Experiments

4.1 Data

Our dataset scraped from `myanimelist.com` (MAL) consists of 132417 ratings for the 10000 top animes, given by 13874 users. We applied WMF on the train preference matrix (13874×8000), excluding users who did not rate any in the train set, to obtain U^{train} and V^{train}.

For each anime, we scraped its MAL ID, numerical features (#average rating, #popularity, rank, #popularity, #members, #favorites, etc.), and all reviews concatenated as one string. We also collected 27266 anime pairs (recommendations). Each (anime 1, anime 2) pair comes with all recommendations written on anime 1's page for anime 2.

4.2 Evaluation method

We used the following metrics:

- Top-K item recall for each user.
- Top-K user recall for each item.
- Mean reciprocal rank for (MRR) each user. (see Appendix A for a discussion on why we chose this over precision-based metrics).
- Mean reciprocal rank for (MRR) each item.
- Qualitative patterns across recommendations.

4.3 Experimental details

4.3.1 Baselines

Throughout our experiments, we set dropout_p=prob_dropout = prob_transform. Our initial experiments set dropout_p=0 to ensure the train loss goes straight down (it should go to zero since the model just needs to learn to output $U_u \dot{V}_v$). This is to make sure the model was implemented correctly.

Our subsequent experiments set dropout_p=1, which forced the model to reconstruct relevances *solely* from content features. We explore different encoders for **anime2vec**.

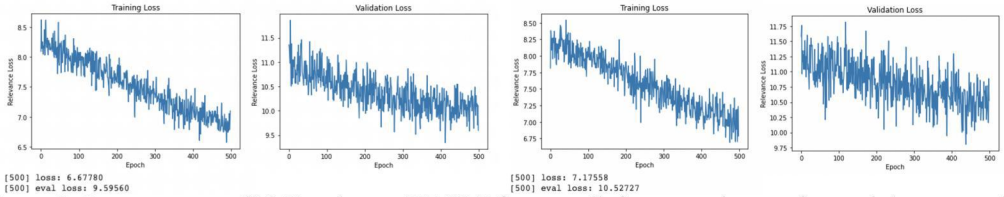

Figure 3: Losses across 500 Epochs, on TF-IDF features (left: synopsis + reviews, right: synopsis + reviews + recommendations), dropout_p=1

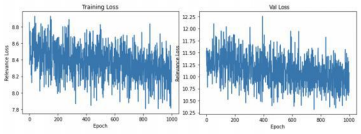

Figure 4: Losses across 1000 epochs for DropoutNet with Word2Vec with single layer modules
For more experiments, see Appendix D

Impressively, these models successfully generalize to the 1000 unseen shows in the validation set with TF-IDF and word2vec features. We also investigated appending to the individual animes all the written recommendations involving it. Interestingly, this increased both bias and variance.

To make better use of the recommendations, we concatenated the encodings to the TF-IDF anime2vec learned graph embeddings using GINEConv.

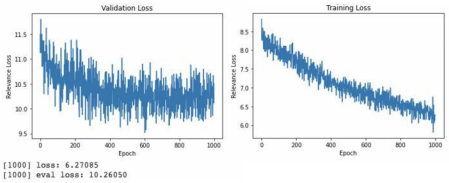

Figure 5: Losses across 1000 epochs, GINEConv, dropout_p=1

For comparison, we ran GCNConv as a baseline, with two initialization methods.

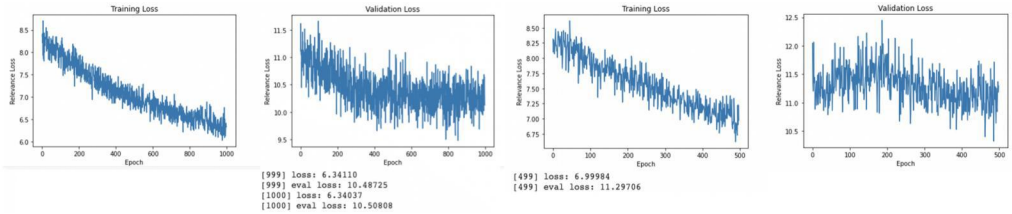

Figure 6: Losses across 1000 epochs, GNNConv, TF-IDF initialization (left) and one-hot initialization (right), dropout_p

4.3.2 Fine-tuning BERT

We ran into significant resource constraints with BERT. Both Colab and Azure limited us to a batch size of 4. Both completed at most 5 epochs overnight. In our milestone, we proposed smarter sampling strategies for efficient training. To address both, we implemented a threshold and sampling strategy that repeatedly sampled a minibatch of (user, item) entires with true relevance (UV^T) outside of a range. We also limit U_ϕ, the user's featurization, to at most their top 5 shows (later, we show this has a clear effect on recall). To smoothen the plots, we took the running loss average.

To motivate DropoutNet, we first ran BERT *without* added modules (just BERT $\rightarrow$ Linear $\rightarrow$ Dot Product $\rightarrow$ Relevances) and dropout_p=1 (so it learns solely off of U_ϕ and V_ϕ).

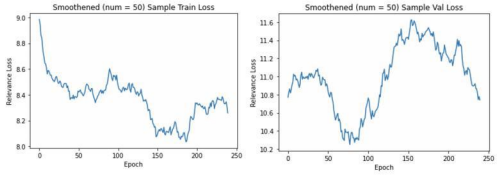

Figure 7: Smoothened losses across 200+ epochs, BERT $\rightarrow$ Linear $\rightarrow$ Dot Product, SGD, dropout_p=1

In Appendix C, we illustrate some additional insights and experiments where we only consider synopsis/reviews.

For comparison, here is static BERT with all layers frozen, trained with the same configuration.

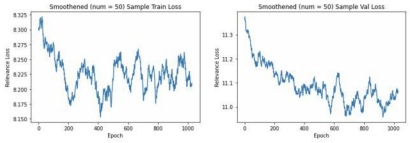

Figure 8: Smoothened losses across 1000 epochs for static BERT (all layers frozen), dropout_p = 1

Next, we fine-tuned the last Transformer Layer 5.

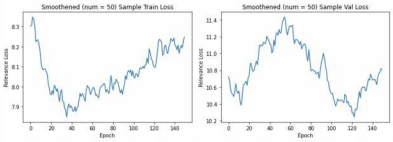

Figure 9: Smoothened losses across 200+ epochs, fine-tuned BERT, batch_size = 4, dropout_p = 1

In Appendix B, we present our theory as to how bias *increases*, variants of these experiments that support it, and more on the difficulty of fine-tuning BERT end-to-end with DropoutNet.

Our most successful experiments took a middle ground by installing a learning rate decay to the *BERT fine-tuned parameters*, while keeping the learning rate for DropoutNet parameters fixed at 1e-4. There are a few more runs (Appendix B) that reveal valuable insights for fine-tuning BERT and the synergy between upstream encoders and downstream layers.

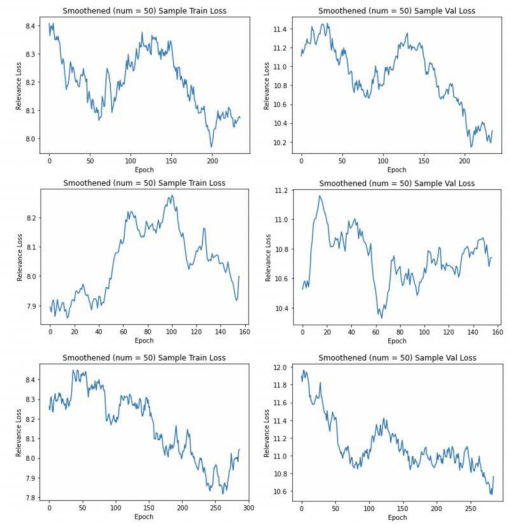

Figure 10: Smoonthened losses for BERT with different exponential decay rates, 0.1 (top), 0.9 (middle), 0.99 (bottom)

4.3.3 Transfer Pre-training BERT

We then *transfer pretrained* off the original pretrained weights from HuggingFace on all text data available across all animes (synopses, reviews, and recommendations) using masked language modeling. We then tested the new static representations on DropoutNet by varying dropout rates.

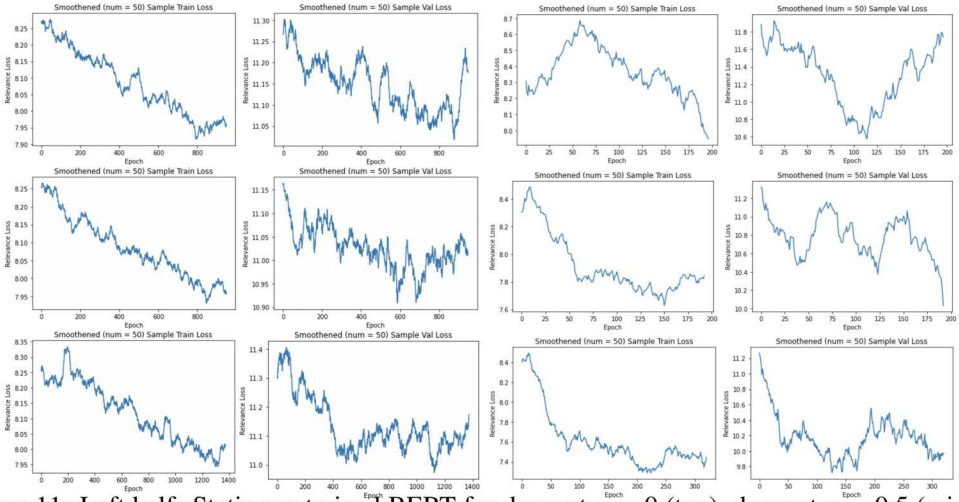

Figure 11: Left half: Static pre-trained BERT for dropout_p = 0 (top), dropout_p = 0.5 (middle), dropout_p = 1 (bottom). Right half: Fine-tuning transfer pre-trained BERT for fixed 1e-4 (top); 1e-4 decay rate = 0.9 (middle); 1e-4 linear decay to 0 at epoch 50 (bottom)

The validation losses ($\approx 11.05, \approx 10.9, \approx 11.0$ for rates $0, 0.5, 1$ respectively) confirm the benefit of a suitable dropout rate for generalization.

We then fine-tuned after in-domain (synopses, descriptions, written recs) pre-training with masked language modeling, with dropout_p = 1. After our insights earlier, we tested different learning rate decay schedules on the BERT fine-tuning params, fixing the learning rate of the DropoutNet parameters at 1e-4.

Fine-tuning was very expensive, but within the first 200 epochs we see the clear benefits of additional fine-tuning, in both lowest validation loss sustained before overfitting and reducing training loss fluctuation across our sampling strategy. We make the observation that a linear decay rate seems better here after in-domain transfer pre-training, whereas an exponential decay was better without it (more in Appendix B).

4.4 Results

We consider 100 items for top-5 user recall, 100 users for top-5 item recall, 100 users for items/user MRR and 100 items for users/item MRR. Due to **significant** computational limits, we only run 10 items and 10 users respectively for BERT models.

Model	Top-5 User Recall	Top-5 Item Recall	MRR (Items/User)	MRR (Users/Item)
TF-IDF	0.58 (0.55)	0.75 (0.57)	2.2e-4 (2.5e-4)	4.1e-4 (1.6e-4)
TF-IDF + GNN	0.54 (0.52)	0.77 (0.55)	2.3e-4	3.6e-4
TF-IDF + Edge-Conv	0.57 (0.56)	0.74 (0.53)	3.2e-4	8.7e-4
Static BERT	0.7 (0.68)	0.67 (0.54)	2.2e-4	**11.5e-4**
Fine-tuned BERT	0.5 (0.47)	0.72 (0.60)	2.5e-4	**8.7e-4**
Static Pre-trained BERT	0.8 (0.75)	0.72 (0.69)	**4.0e-4**	2.6e-4
Fine-tuned Pre-trained BERT	0.6 (0.68)	0.72 (0.69)	**5.2e-4**	2.3e-4

Note 1: The expected result of *random guessing* is in parentheses. It remains the same across models for MRR. We kept the threshold at 0.0 the same across the BERT models, but acknowledge that is biased towards positive samples due to our sampling strategy. Note 2: For the recalls, we threshold TF-IDF baselines, which were trained with a sigmoid layer, by 0.0. We threshold BERT models, which were trained without it, by 0.15. For full recall plots, see Appendix A.

5 Analysis

All things considered, all models convincingly outperform random guessing. TF-IDF seems to outperform on item recall (which may be due to confounds like the thresholding; see Appendix A). However, each BERT model outshines TF-IDF in at least one MRR department: items/user MRR for the pretrained models and users/item MRR for the non-pretrained. It seems that in-domain transfer pretraining significantly boost the rank of a user's top show, at the cost of ranking a show's most relevant customer.

In our app, we compare **DeepAniNet** with two traditional approaches that relied only on the preference matrix: WMF and top-K collaborative filtering.

As an example, our app (currently only hosting the top-235 shows used for our milestone) produces the following when querying with two highly regarded shounens ["Fullmetal Alchemist Brotherhood" (FMAB), "Shingeky No Kyojin" (AoT/Attack on Titan)]:

- WMF: 1) Haikyuu, 2) Hunter x Hunter
- Top-K: 1) Mushishi, 2) Stein's Gate, 3) Haikyuu, 4) Hunter x Hunter
- DeepAniNet: 1) Hajime no Ippo, 2) Nana, 3) Gintama

Below is Some context for the animes:

- FMAB is rated #1 on MAL all-time and AoT is trending #1 in popularity.

- Hunter x Hunter has a huge fanbase overlap with FMAB.
- Stein's Gate and Mushishi are not shounen but highly regarded (especially Stein's Gate).
- Hajime no Ippo is an underrated shounen-of-shounen's.
- Gintama is an offbeat fantasy/satirical/historical "shounen".
- Nana, a lesser known slice of life, has a synopsis that says two girls travelling together in search of one girl's boyfriend (important: FMAB's synopsis is about two brothers travelling together in search of the philosopher's stone).

Our Appendix F section contains further qualitative insights. We make the following generalizations: WMF outputs whatever is popular; Top-K outputs what a selected set of others enjoyed; while **DeepAniNet** seems capable of *actually reading* information on other shows and outputting underrated but thematically similar shows.

6 Conclusion

We first replicated DropoutNet's findings on a novel domain and dataset - anime recommendations - which we built our own data discovery, scraping, and processing pipelines for. With even basic TF-IDF features, we demonstrate generalization to cold start shows. We then incorporated in the graph structure of MyAnimeList's recommendations tab via trainable, deep GNN layers that showed superior metrics. We then turn to BERT for pre-training and fine-tuning deep representations for DropoutNet as a downstream task.

We demonstrated DropoutNet is not only necessary, but a well-chosen dropout_p and fine-tuning strategy (such as in-domain pre-training and learning rate decay) is essential to unlocking BERT's encodings for reconstructing relevance scores. During our bootstrapped journey, we had to impose major some major handicaps on BERT:

- A maximum batch size of 4 due to GPU memory.
- A threshold and small sampling strategy due to RAM memory.
- Limiting user featurization to just 5 shows due to GPU cache memory.
- Limiting passage length to 512 due to BERT encoding scheme limit.

Nonetheless, BERT still defeated our baselines' best validation loss before overfitting (≈ 9.8), which substantiates our original hypothesis: modern pre-trained encoders deliver superior performance than TF-IDF, the SOTA choice at the time of DropoutNet's authors, for cold-start generalization on content recommendations.

We also suffered from inherent difficulties in training large-scale recommendation systems, including sparse preference data, biased/non-i.i.d. data generating distribution, and other inconsistencies across users and items. We overcame some of these difficulties by building our own show and user discovery, scraping and preprocessing pipeline. Rerunning and retraining with different configurations and conducting more hyperparameter tuning can be realistic next steps to improve performance.

On the deep learning side, if we were not limited by compute, deeper architectures and more hyperparameter (especially dropout_p and transform_p) sweeps, or even just sampling and fine-tuning longer, would have certainly helped. Two future directions seem particularly promising:

- Further investigating the relationship between upstream pre-training and downstream layers and how to better train an end-to-end system (see more in Appendix B).
- Self-supervised or unsupervised techniques (such as knowledge graph, entity pre-training; see more in Appendix E) for more robust anime2vec representations in downstream training.

With these in mind, we plan to continue this project after the course, investigating the aforementioned approaches and hopefully spinning this off into something much more than what we submitted.

References

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[2] Andrzej Cichocki and Anh-Huy Phan. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 92(3):708–721, 2009.

[3] Cédric Févotte and Jérôme Idier. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural computation*, 23(9):2421–2456, 2011.

[4] Alberto García-Durán, Roberto Gonzalez, Daniel Oñoro-Rubio, Mathias Niepert, and Hui Li. Transrev: Modeling reviews as translations from users to items. In Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins, editors, *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part I*, volume 12035 of *Lecture Notes in Computer Science*, pages 234–248. Springer, 2020.

[5] Prem Gopalan, Jake M Hofman, and David M Blei. Scalable recommendation with poisson factorization. *arXiv preprint arXiv:1311.1704*, 2013.

[6] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[7] HuggingFace. Fine-tuning a language model.

[8] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[9] Aäron Van Den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Neural Information Processing Systems Conference (NIPS 2013)*, volume 26. Neural Information Processing Systems Foundation (NIPS), 2013.

[10] Maksims Volkovs, Guang Wei Yu, and Tomi Poutanen. Dropoutnet: Addressing cold start in recommender systems. In *NIPS*, pages 4957–4966, 2017.

[11] Chong Wang and David M Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456, 2011.

[12] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1235–1244, 2015.

[13] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 495–503, 2017.

[14] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval, 2019.

[15] Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model, 2019.

[16] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. ERNIE: enhanced language representation with informative entities. *CoRR*, abs/1905.07129, 2019.

A Appendix A: Additional Experiment Details + Visualizations

As promised, we include plots for user/item recall. For reference, we plot the baseline of *random guessing*, preconditioned by the model producing the same number of predictions, for our best BERT models.

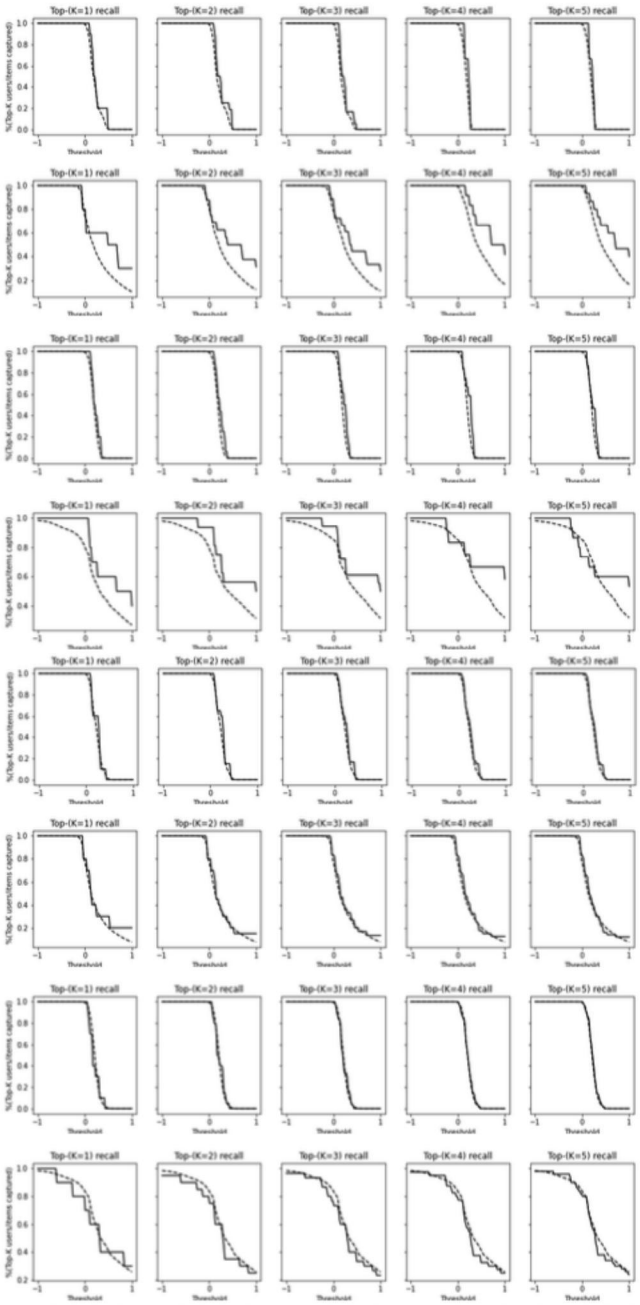

Figure 12: Upper Half: User Recall, Lower Half: Item Recall; Rows 1-4 of each half: static, fine-tuned, transfer pre-trained static, transfer pre-trained and fine-tuned

Overall, BERT still easily beats random guessing in recalling items/users but pales compared to TF-IDF baselines. As our plot hints, the threshold is different from that of TF-IDF. This is a undesiable byproduct of our sampling strategy, where we chose asymmetric negative and positive cutoffs of -6.0 and 10.0 for a comparable number of positive and negative samples.

While top-k recall tests how well the model can retrieve the top-relevant users/items, top-k precision is a worse metric due to the inherent sparsity of the preference matrix. It is unfair to penalize the model's confident predictions, since most users have probably not watched most of the shows they can enjoy, especially since our BERT models were trained from a sampling strategy that had a near-even split between highly favorable and unfavorable shows. After discussing with our mentor, we went with a more reasonable alternative - Mean Reciprocal Rank - which looks at the entire ranked list of predictions for the most relevant item and doesn't complicate itself with further relevant items.

B Appendix B: Downsides of End to End Training

As mentioned in the Analysis, we believe the reason fine-tuning BERT along DropoutNet led to an ascending training loss is due to domain shift (from DropoutNet's perspective). In other words, DropoutNet is always one step behind BERT's changing representations.

Our explanation is made more nuanced by the non-pretraining results, where we tried decaying rates of $0.1, 0.9, 0.99$ for *BERT fine-tuning params*.

- Learning rate decay with exponential rate 0.1: We saw from Figure 8 training loss initially decrease, then the model becoming becomes incapable of further reducing bias. The noise introduced from sampling result in a gradually increasing training loss. Similarly, Figure 10 shows the same thing happen, except the model recovers $\approx$ epoch 120. It seems even a few initial epochs of fine-tuning helped, before it becomes essentially static BERT.
- Learning rate decay with exponential rate 0.9: We see the domain shift theory in action, with training loss increasing until $\approx$ epoch 110, when DropoutNet adapts to the shifted representations. Validation loss increases throughout, so it seems fine-tuning here hurt more than helped.
- Learning rate decay with exponential rate 0.99: This was the only rate with consistently decreasing training and validation loss curve. It seems the domain shift occurs slowly, allowing both BERT and DropoutNet to adaptively train.

We ran additional experiments to test our hypothesis. First, we did an experiment where we froze the BERT fine-tuning layer at epoch 50.

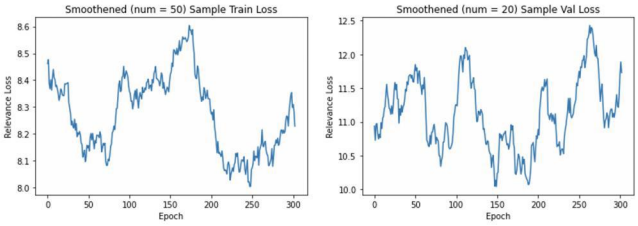

Figure 13: lr 1e-4, freeze BERT fine-tuning params at epoch 50

At epoch 50, bias spikes but variance goes down to 10.5 until around epoch 175 when it overfits again. Now, what if we use a higher learning rate for BERT while fixing DropoutNet at 1e-4?

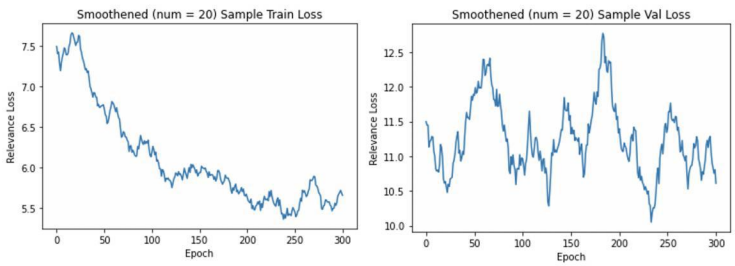

Figure 14: lr 1e-3 for BERT fine-tuning params, 1e-4 for DropoutNet

As you can see, this better reduces bias but fluctuates on the validation set at 11.0 the whole way. It seems our theory is substantiated, and we would love to learn more about the research area of gradual domain adaptation and its effect on downstream and end-to-end to training.

C Appendix C: Reconstructing relevances directly from BERT

In lieu of the difficulties training our system end-to-end from Appendix B, we wondered if DropoutNet's autoencoder-denoising inspired architecture was even necessary for reconstructing relevances. We ran experiments following Figure 7 on solely synopsis and solely reviews data.

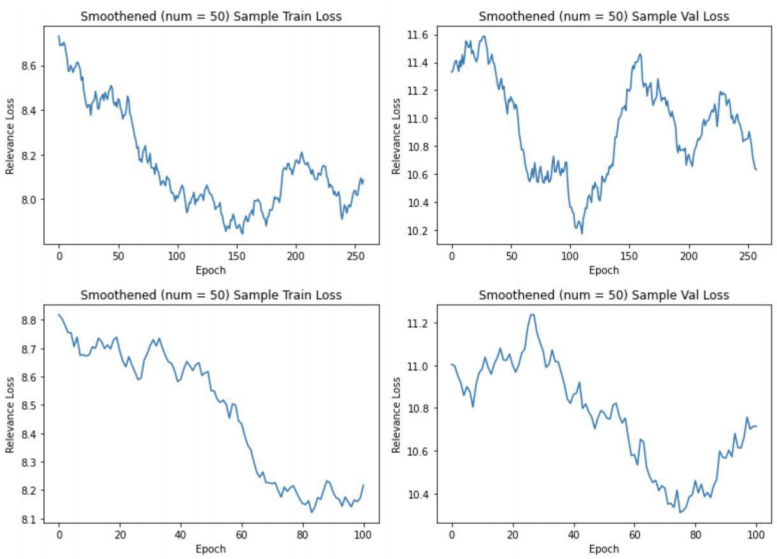

Figure 15: Figure 7, except only on synopsis (left) or reviews (right) only data

D Appendix D: Experiments with the depth of f_U, f_V, f_{ϕ^U}, f_{ϕ^V}, $f_{\mathcal{U}}$, and $f_{\mathcal{V}}$

In the original DropoutNet paper, only single layer modules were used. Thus, previous to this point the functions f_U, f_V, f_{ϕ^U}, f_{ϕ^V}, $f_{\mathcal{U}}$, and $f_{\mathcal{V}}$ have all been single layer linear layers (projection layers), the minimum requirement needed for DropoutNet to work. We inquire whether deeper, more sophisticated modules would improve performance. Experiments were done on a DropoutNet with word2vec feature encoder, by replacing each projection layer with a 2-layer module, 3-layer module, 2-layer module with batch normalization, 2-layer module with dropout, and 2-layer module with both batch normalization and dropout.

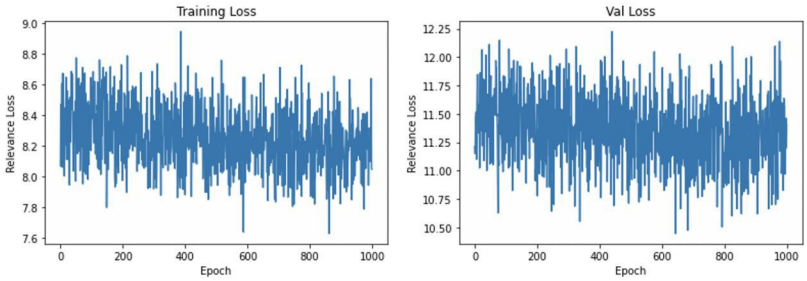

Figure 16: Double layers, dropout$_p = 1$

Adding layers to DropoutNet reduce its ability to generalize. This was another early sign we needed deeper representations than shallow features like TF-IDF and word2vec to reduce bias.

Table 1: Training and Validation losses of DropoutNet with various modules.

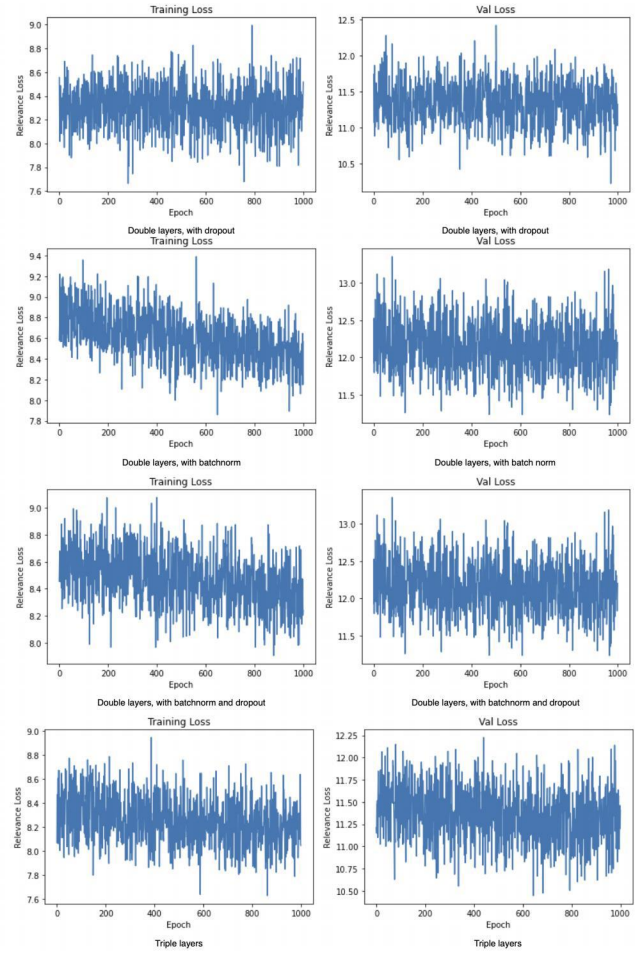

Figure 17: Double layers with dropout, double layers with batchnorm, double layers with batchnorm and dropout, and triple layers loss plots

We find a surprising result: it seems that the double-layer module only had marginally better performance than the single-layer module, and the triple-layer module actually performed worse than the single-layer module or the double-layer module. This might be because multi-layer modules take longer to train. However, there is an plateau in validation loss for triple layer modules, which implies otherwise. In any case, the advantages of deeper modules may be nullified as the tasks of f_U, f_V, f_{ϕ^U}, f_{ϕ^V}, $f_{\mathcal{U}}$, and $f_{\mathcal{V}}$ should be relatively simple.

E Appendix E: Graph Structure Integrating Knowledge to Anime2Vec

Our experiments demonstrated appending written recommendations to an individual anime's encoding text didn't boost performance for either TF-IDF or BERT. We incorporated written recommendations in a more effective way via GINE (edge conv), which demonstrated better results (only showed for TF-IDF; not in time for BERT). However, the graph is still extremely sparse, and the trainable vertex embeddings are difficult to evaluate intrinsically.

Thus, we believe there to be more interesting ways to pre-train on written recommendations for more robust, knowledge-integrated representations.

We were particularly inspired by Megan Leszczynski's lecture. We effectively already use Transformer encodings[14] for our entity embeddings. It also requires little additional effort to add masked entity token prediction used in ERNIE's[16] pretraining objective. Additional papers (some cited in bibliography) whose entity/knowledge pretraining techniques can potentially be tested. Finally, techniques in salient span masking[15] that involve altering the training data (such as removing mentions of animes) have been shown to be a cheap yet effective way for better transfer pre-training.

F Appendix F: Additional Qualitative Observations + Adversial Examples

Perhaps the biggest advantage of DeepAniNet comes from the fact it actually reads all available information word-for-word pertaining to shows (more precisely, just 512 tokens) to make recommendations. This comes with the obvious benefit of not being biased towards popular anime and the capability of discovering underrated but themetically similar shows. However, it also comes with high sensitivity to certain word-level mentions. An example is when running the query set ['Fullmetal Alchemist Brotherhood', 'Attack on Titan'] (FMAB and AoT), two highest rated/regarded shounen series of all time.

Model	Top-5 Recommendations
TF-IDF GNN (epoch 999)	'A Farewell to Arms', 'Armored Trooper Votoms', 'Hotori: Tada Saiwai wo Koinegau', 'Sailor Moon SuperS the Movie: Black Dream Hole', 'Bubblegum Crisis'
TF-IDF GINE (epoch 999)	'Armored Trooper Votoms', 'Bubblegum Crisis', 'A Farewell to Arms', 'Slayers', 'Patlabor: The Movie'
Fine-tuned BERT (epoch 200+)	'Royal Space Force: The Wings of Honneamise', 'Desert Punk', 'Ginga Eiyuu Densetsu: Die Neue These - Seiran 3', 'Mobile Suit V Gundam', 'Azur Lane'
Fine-tuned pre-trained BERT	'.hack//Sign', '.hack//Quantum', 'Fate/Zero', '.hack//Liminality', '.hack//Gift'

For some context:

- A Farewell to Arms: a story of "power suit-wearin' men tasked with disarming automatic tanks in a post-apocalyptic Tokyo"
- Armed Trooper Votoms: set in "a century of bloodshed between warring star systems... flames of war..." where "a special forces powered-armor pilot is suddenly transferred into a unit engaged in a secert and highly illegal mission to steal military secrets..." (you get the idea)
- Hotori: "At the Personality Plant, robots are being built and slowly outfitted with the artificial memories of real people." The main character, Suzu, "is one such robot."
- Bubblegum Crisis: set in a city "built form the labors of mechanical beasts... with incredible destructive power as a new type of advanced weaponry"
- Azur Lane: pits "a divided humanity" which "stood in complete solidarity" against "an alien force with an arsenal far surpassing the limits of current technology"; with countries joining forces, "paving the way for the improvement of modern warfare"... during "neverending conflict within humankind" (basically if FMAB and AoT had a baby... this would be it!)

It thus seems GNN/GINE significantly improve results and converge on a similar set of results that all share a common set of themes: military, humanity's war, revolution, etc. Reading FMAB and AoT's descriptions, we see these themes repeated verbatim in their synopses: "military allies, colonel, lieutenant, nationwide conspiracy, state, law" and "humanity, extinction, defensive barriers, fight for survival, Survey Corps, military unit, brutal war, walls". We also see more plot-level similarities with FMAB and Hotori. FMAB is about two brothers who lost parts of their physical bodies. "It is the hope that they would both eventually return to their original bodies that gives Edward the inspiration to obtain metal limbs..." whereas Hotori's synopsis strikes a similar theme.

TF-IDF seems capable of capturing similarities from narrow dimensions (such as mentions of military themes), even if those dimensions are not the main focus of either FMAB or AoT.

The fine-tuned BERT, meanwhile, **captures similarity across more complex dimensions**. For some context on its recommendations:

- Royal Space Force: Protagonist is part of the country's space force, who embark on a mission to redeem humanity by restoring its strength
- Ginga Eiyuu Densetsu: About a coup staged by the National Salvation Military Council under the direction of the Galactic Empire, happening during civil wars in both the Alliance and the Empire
- Mobile Suit V Gundam: About a space immigrant who joins the League Militaire, a militia frustrated with their empire's cruelty, who fights to bring an end to the Zanscare Empire's reign

The shows are more diverse in plot while rooted in common themes across militant conflict, failure of government, humanity, and revolution.

Post pre-training, it seems BERT "overfitted" to in-domain text and fell prey to this adversial example. .hack is a franchise consisting of multiple shows, whose MyAnimeList reviews' page is full of mentions of the franchise. Pre-training must've seen ".hack" way too often and built near-identical representations for each show in the franchise, causing the four of them to be recommended together.

Looking past that, Fate/Zero is a *very* good recommendation that's not only thematically similar (war, battle royale, etc.) but is regarded as a crossover between both FMAB and AoT: exploration of deep themes and unapologetic cruelty.

At the same time, many recommendations are harder to understand. While it's fascinating to try rationalizing these outputs one-by-one and try to find common patterns, it'd be more intuitive if we can have the model select the relevant terms that had the highest activation. This could be an intriguing avenue of future research, and one of most interest to the anime community!