Autoencoders for Research Paper Recommendation System

Keshav Worathur
University of Toronto
Toronto, ON M5S 1A1
keshav.worathur@mail.utoronto.ca

Abstract

Recommendation systems have enjoyed widespread adoption in online commerce, news websites, and social media. Building recommendation systems is challenging because high-dimensional data, such as article titles, need to be compressed into forms that computers can process. We propose a recommendation system for research papers that uses a paper's title to recommend similar papers. To achieve this, we develop an unsupervised clustering algorithm on research paper titles using the autoencoder architecture. In particular, we use a recurrent neural net (RNN) to "encode" paper titles as low-dimensional embeddings in latent space. We condition the autoencoder's latent space geometry by reconstructing paper titles from their embeddings with a "decoder" RNN and back-propagating the reconstruction error signal through the network to minimize distances between structurally similar titles. Further, we aim to condition latent space geometry to minimize distances between conceptually similar titles by modifying the baseline model. The resulting model, called the Denoising Adversarial Autoencoder (DAAE), incorporates a denoising training objective by training with noisy training data and an adversarial training objective by imposing a prior distribution on its latent variables. We compare this DAAE model with the baseline autoencoder model by performing k-nearest neighbor searches in their latent spaces, simulating real-world usage of the recommendation system. Based on our findings, we determine that while the DAAE represents an improvement over the baseline autoencoder, the recommendations are primarily helpful for junior researchers who have not established a focused research sub-field for their searches.

1 Introduction

This section details the motivations for our research, our proposed method, and principal objectives. Our work is motivated by researchers' challenges when finding work related to their research questions. Currently, research paper databases support limited search criteria such as searches by title keywords and author names (1). Using these criteria effectively assumes a familiarity with existing work in the field of interest (2). While survey papers can help develop familiarity with existing work, they typically do not reference all works in a research area due to constraints on the publication period for surveyed works and/or search phrases used during the survey.

Recognizing limitations of existing methods for discovering works relevant to a focus area, we devise a system that provides tailored, "more like this" (MLT) - style recommendations as researchers refine their search. This system can make the literature search more productive by helping researchers prioritize papers most relevant to their focus. For example, in the field of computer science, research papers typically fall under one of three categories: "theoretical" (e.g. proving a theorem), "engineering" (e.g. programming and analyzing a system), and "empirical" (e.g. testing hypotheses through experiments). A researcher studying "vector databases" may be most interested in "engineering" papers that conduct performance profiling of these systems. In this example, the recommendation system can adapt to the researcher's search direction and recommend more "engineering" papers.

Our recommendation system uses a query title and a database of reference titles to recommend similar papers to a user. When building the reference database, we aim to categorize similar papers. To flexibly categorize papers similar to the example in the previous paragraph, we train an unsupervised clustering algorithm where category labels are not specified prior to clustering. To perform title comparisons that go beyond keyword searches and use higher-order concepts present in a title, we use a deep learning algorithm to perform clustering. In particular, we use an autoencoder with recurrent neural networks (RNNs) for the encoder and decoder networks, with the encoder network clustering similar paper titles as it maps them to the model's latent space. As text data is complex, with sentences of varying length, structure, and word choice communicating the same idea, using non-linear functions for the encoder and decoder allows the model to learn meaningful representations of paper titles.

These representations, known as embeddings, are lists of floating point numbers that support the use of arithmetic operations to measure the similarity between research paper titles. When building the reference database, paper titles are fed through the encoder network of the autoencoder, which will map the titles to a low-dimensional latent space where they are represented by embeddings. When the system is providing recommendations, input titles will be mapped to the latent space and their k-closest embeddings (measured by a metric such as euclidean distance) will be mapped back to titles in the reference database and displayed to the user. In a recommendation system, it is important for embeddings to be of low dimension, allowing for efficient comparison, while retaining enough information in paper titles to allow for relevant recommendations.

We argue that the autoencoder family of models is well-suited at balancing these two goals. The autoencoder consists of a bottleneck layer with a fixed number of hidden units that is smaller than the dimension of the input layer, allowing for data to be compressed into more manageable forms (3). With the dimension of the bottleneck layer being fixed, the autoencoder ensures that embeddings still preserve key properties of the input data through using training objective that aims to minimize the reconstruction loss of the decoder network. Specialized models such as Denoising Adversarial Encoders (DAAE) proposed by Shen et. al (4) have been shown to preserve the semantics of text data through added denoising and adversarial training objectives.

To undertake our research we outlined three principal research objectives. First, we aim to build a recommendation system for research papers to help researchers quickly find works related to a given paper. Second, we wish to study the autoencoder model and DAAE models in greater depth. Finally, we hope to understand the limitations of encoding low-dimensional embeddings of high-dimensional data. Together, these objectives allow us to develop title embeddings as the foundation of a research paper recommendation system. For our implementation, we use the autoencoder library developed by Shen et. al (4) which is made available by the authors at https://github.com/shentianxiao/text-autoencoders.

The paper is divided into multiple sections to present the ideas contained within. First, we introduce prerequisite background material and discuss related works. Then, we will explore a curated dataset of research paper titles which will be used to train the AE and DAAE model. The autoencoder architecture and its relation to the recommendation system is discussed in detail. We perform k-nearest neighbor searches in the latent spaces constructed by the two models to generate recommendations, which we evaluate qualitatively. We remark on the desirable properties of the DAAE model and discuss the model's limitations. Finally, we discuss ethical considerations when building recommendation systems for scientific articles.

2 Background and Related Work

Recommendation systems help personalize internet users' experiences by suggesting content that aligns with their interests. These systems can be categorized as content-based, collaborative filtering, or hybrid systems, depending on the data used for recommendations (5). Content-based recommendations systems typically recommend items by matching item descriptions to a user's interests (also known as a user profile) or previous items the user interacted with (6). While content-based systems consider the user's behaviors in isolation, collaborative filtering systems consider the behaviors of similar users, using their user-item interactions to provide recommendations (5). In this paper, we study content-based recommendation systems to help address the challenges underscored in Section 1.

We now provide a brief overview of previous work on content-based recommendation systems for academic work, highlighting the diversity of input data and document representations used for recommending papers. Starting with input data, many content-based approaches may require the name of a paper or keywords to provide recommendations, while others use observations of a user's behavior over time (2). A content-based system also must specify a numerical representation of documents in order to compute similarity. Document representations exploit diverse combinations of metadata (e.g. title, abstract, authors, keywords) (7), citation relationships (8), and the full article text (9). Once the recommendation system has determined the most similar documents to the query paper/keywords/user, it will typically present a list of recommended papers. To measure performance of content-based recommendation systems for research, benchmark datasets such as the scholarly papers recommendation dataset (SPRD) have been introduced (10). A common evaluation metric for measuring recommendation quality is the mean reciprocal rank (MRR), first introduced by Ricci et. al in (11) .

A challenge in content-based systems is that the large corpora of document information needs to be distilled for real-time recommendations from databases of hundreds of thousands of documents. Common methods for document distillation include term frequency - inverse document frequency (TF-IDF) and document embeddings (2), with our method falling under the latter category. The goal of document embedding is to capture the most important features of multi-page documents in a low-dimensional vector. To this end, a majority of previous works focused on embedding unstructured document text (12; 13) i.e. document data. However, works focused on embedding document metadata are less common. For example, Guo et. al used a word2vec model to learn embeddings of a paper's abstract, where each sentence in the embedding was weighted based on it's relevance to the paper's title (14). The authors found that their method achieved improved MRR compared to other state of the art methods (14).

We corroborate Guo et. al's results by using deep learning models to learn embeddings using only a research paper's title. To our knowledge, no previous works have relied on a single piece of document metadata to recommend research papers. We chose this approach because titles are written to succinctly capture a paper's main idea and therefore serve as rich sources of document data. Through our work we hope to assess whether titles alone encode sufficient information about a document to provide relevant recommendations to a user.

To learn embeddings of paper titles, we choose the autoencoder family of models (15). The autoencoder architecture consist of a encoder which maps input data to a latent vector, also referred to as a an embedding. The latent vector is then mapped back to the data space using a decoder, which aims to reconstruct the original data from the latent vector with minimal error. The encoder acts as a clustering algorithm, grouping titles that share a common theme close together in the latent space constructed by the encoder. While the idea of making recommendations via clustering is not novel (16; 12; 13), we aim to improve thematic clustering of ideas using an enhanced model called the denoising adversarial autoencoder (DAAE) proposed by Shen et. al (4). This model introduces denoising and adversarial training objectives, which Shen. et al attribute to improved mapping of similar sentences to nearby latent vectors in the latent space of the autoencoder. We perform ablation studies to qualitatively determine whether these additional training objectives improve the recommendations provided by our system.

3 Data

This section will introduce our selected dataset and explore various properties of the data that contribute to the difficulty of the problem. We chose to use the arXiv dataset (17) on Kaggle which includes metadata for 1.7+ million scholarly articles. Included in the document metadata are the title, authors, and abstract. Our dataset consists of all 200,640 titles under the "Artificial Intelligence" and "Machine Learning" categories. Furthermore, there are 61, 287 unique words across these titles. We randomly allocated 160, 640 titles for training, 20, 000 for hyperparameter tuning, and set aside 20, 000 for our evaluation of model performance.

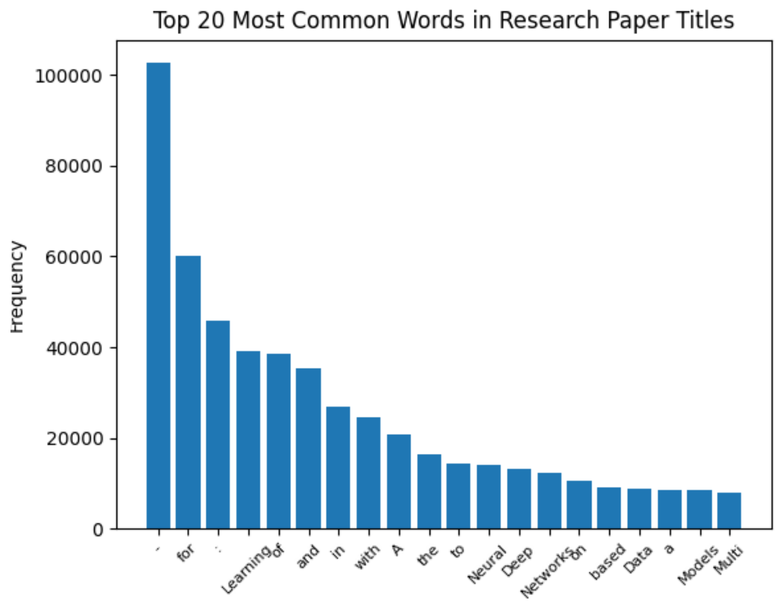


Figure 1: Distribution of different words in the dataset

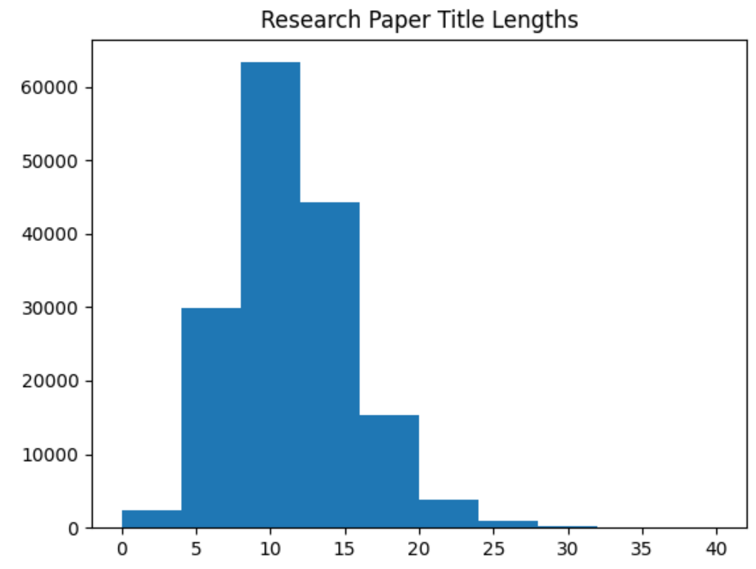


Figure 2: Histogram of number of words per research paper title

Observe that in figure 1 there is a long tail of infrequently occurring words in the dataset. Learning embeddings for rare words will increase the model's size significantly while not producing an equal improvement to embedding quality. Given the large number of unique words we limit the set of words the autoencoder learns to represent, which we call the vocabulary. We decided to restrict the vocabulary size to the top 10,000 most frequently occurring words. In figure 2, we can observe that the majority of titles are between 5 to 15 words in length. The scarcity of longer titles may result in reduced quality of embeddings for these titles, which we address in Section 7.

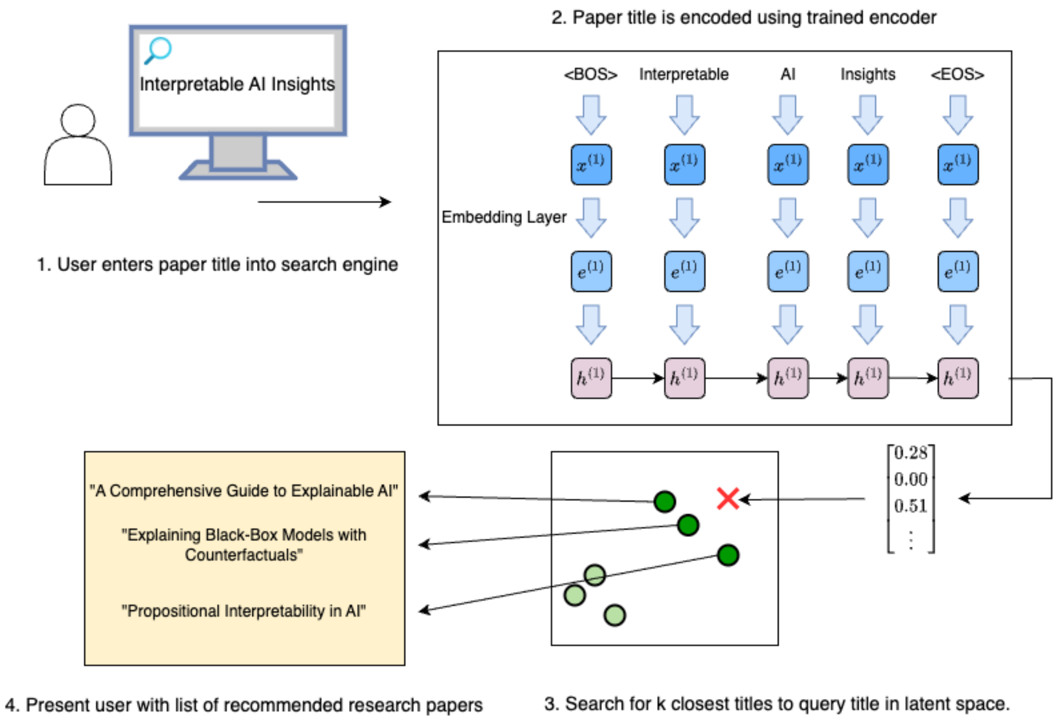
To prepare the input data, we use a process called tokenization, which converts text sequences to numerical sequences. A token is a unique integer identifying a word in the vocabulary. For each sequence, we prepend the beginning of sequence token `<bos>` and append the end of sequence token `<eos>`. To represent words that are not in the vocabulary we add an `<unk>` (unknown) token to our vocabulary.

In this section, we have explored our chosen dataset for training the autoencoder and described the pre-processing steps taken to prepare the data for training. We discussed the process of tokenization, which is a crucial step in pre-processing data for any ML model trained on text data. In section 4, we will show how the tokens are provided as input to the model to learn embeddings of titles.

4 Methods

4.1 Autoencoder

The model's architecture consists of a pair of networks: the **encoder** and the **decoder**. Figure 3 describes how the encoder network is used in the recommendation system.



Interpretable AI Insights
1. User enters paper title into search engine
2. Paper title is encoded using trained encoder
<BOS>
Interpretable
AI
Insights
<EOS>
$x^{(1)}$
Embedding Layer
$e^{(1)}$
$h^{(1)}$
0.28
0.00
0.51
"A Comprehensive Guide to Explainable AI"
"Explaining Black-Box Models with Counterfactuals"
"Propositional Interpretability in AI"
4. Present user with list of recommended research papers
3. Search for k closest titles to query title in latent space.

Figure 3: Steps for obtaining paper recommendations using the trained encoder.

Encoder. The encoder is displayed in the second step of figure 3. The encoder encodes the input title in steps, updating an underlying representation $h^{(t)}$ of the title over time. At each time step t, the next token in the sequence is passed down into the embedding layer. The embedding layer uses the one-hot vector representation of the token to select a row from a $V \times D_{\text{emb}}$ matrix storing token embeddings, where D_{emb} is the size of each token embedding. Next, the embedding $e^{(t)}$ is passed down as to a Gated-Recurrent Unit (GRU) which updates the hidden state $h^{(t)}$, a vector of size D_h, using $e^{(t)}$ and the previous hidden state $h^{(t-1)}$. This process repeats until the entire sequence of length T has been encoded. The hidden state $h^{(T)}$ is fed through the final layer of the encoder to obtain a compressed latent vector z, which will capture information about the entire title.

Decoder. The decoder reconstructs the input title in steps, updating a hidden state which influences the words that are added to the reconstructed sequence. The hidden state is initialized using the latent vector z and the `<bos>` token, marking the first token of the sentence. At each time step t, the decoder outputs a probability distribution $m^{(t)}$ over the vocabulary, computed using equation (1):

$$\mathbf{m}^{(t)} = \phi\left(\mathbf{U}^\top \mathbf{m}^{(t-1)} + \mathbf{V}^\top \mathbf{x}^{(t)} + \mathbf{b}\right) \tag{1}$$

Equation (1) shows that the previous hidden state $\mathbf{m}^{(t-1)}$ is combined with the current input $\mathbf{x}^{(t)}$ to update the current hidden state. The combination of $\mathbf{m}^{(t-1)}$ and $\mathbf{x}^{(t)}$ is passed through the softmax activation denoted by ϕ in equation (1). The output represents a distribution over all words in the vocabulary. We sample a word from this distribution to obtain the input to the next time step $\mathbf{x}^{(t+1)}$, making the decoder an auto-regressive model. This process is repeated until the end of the string token is generated or the sequence reaches a cutoff length.

As the sequence progresses, $\mathbf{m}^{(t)}$ is used to make predictions about what word comes next. In this sense, $\mathbf{m}^{(t)}$ encapsulates information about the remaining words to be generated, contributing to the model's anticipation of future elements in the sequence. The update of the hidden state allows the RNN to capture information from both the previous hidden state (which encodes historical context) and the current input word. This updated hidden state becomes the memory for the current time step, influencing the generation of the next hidden state at the time step.

Training Algorithm. Note the generated sequence and input sequence are both discrete, making training with standard gradient descent algorithms infeasible. However, we can cast training as a continous optimization problem by comparing the sequence of continuous *logits* produced by the decoder with the input sequence. Logits are the un-normalized values passed to the softmax activation ϕ in equation (1). Computing the cross-entropy loss between these two sequences gives reconstruction loss defined in equation (2):

$$\mathcal{L}_{\text{rec}}(\theta_E, \theta_G) = \mathbb{E}_{p_{\text{data}}(x)}[-\log_{p_G}(x|E(x))] \tag{2}$$

As expected during the early stages of training, the logits of the decoder will produce random probabilities, causing generated tokens to diverge from the ground truth. Due to the auto-regressive nature of the decoder, this phenomena can have cascading effects on training. Thus, we ground training using a technique called **teacher forcing**. During training, instead of sampling the next word based on the model's distribution $\mathbf{z}^{(t)}$, the ground truth word $\mathbf{x}^{(t)}$ is used in the next step. This stabilizes training by providing more accurate and reliable gradients, especially in the early stages of learning. Hyperparameters used for training are defined below in table 1:

Table 1: Hyperparameters for autoencoder model.

Parameter	Symbol	Value
Vocabulary Size	V	10000
Embedding Size	D_{emb}	512
Hidden Layer Size	D_h	1024
Latent Vector Size	D_z	128
Learning Rate	α	$1e^{-4}$
Batch Size	B	32
Nr. of Epochs	N	20

4.2 Denoising Adversarial Autoencoder (DAAE)

We obtain a DAAE model by adding a denoising and adversarial training objective to the autoencoder loss function in equation (2), resulting in the DAAE loss function in equation (3) below:

$$\mathcal{L}_{\text{DAAE}}(\theta_E, \theta_G, \theta_D) = \mathcal{L}_{\text{rec}}(\theta_E, \theta_G) - \lambda \mathcal{L}_{\text{adv}}(\theta_E, \theta_D), \text{ where} \tag{3}$$

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{p(z)}[-\log(D(z))] + \mathbb{E}_{p_{\text{data}}(x)}[-\log(1 - D(E(X))] \tag{4}$$

First, notice that there is no explicit denoising loss function in equation (3). Instead, we incorporate denoising into model training by adding noise to the encoder's input data, using a data augmentation procedure followed by Shen et. al (4). We add noise by randomly dropping words, randomly "blanking" out words, randomly substituting words with another word from the vocabulary, and shuffling words within a close vicinity of each other. The amount of noise added is controlled by the parameters defined in table 2. As a result of minimzing $\mathcal{L}_{\text{rec}}$ on pre-augmentation data, we implicitly train the autoencoder to denoise noisy inputs.

On the other hand, the adversarial training objective is explicitly included in equation (3) and its expanded form is given by equation (4). To add this training objective, we train a discriminator network D. In our model, D is a single layer neural network with nonlinear activations. The discriminator D classifies whether a latent vector was sampled from the autoencoder's latent space or a prior distribution $p(z)$, which is typically guassian. In other words, the discriminator D is a binary classifier. The discriminator's loss function $\mathcal{L}_{\text{adv}}$ is the cross-entropy loss over inputs from the prior distribution $p(z)$ (first term of equation (4)) and inputs from the encoder's latent space (second term of equation (4)).

During training, the DAAE must balance both objectives of (1) minimizing reconstruction loss and (2) maximizing discriminator loss, resulting in an adversarial training algorithm. When the DAAE optimizes objective (2), a tight coupling is created between the encoder and decoder networks. This coupling arises because the latent variables z produced by the encoder follow a continuous distribution $p(z)$ which the decoder must learn to model (4). This coupling enhances the decoder's ability to generate coherent text that preserves style and high-level syntax. The weight given to objective (2) is controlled by the parameter λ in equation 3. For our use case, we prioritize reconstruction accuracy over the decoder's ability to act as a text generator, assigning a value of $\lambda < 1$. Table 2 below summarizes the hyperparameter settings for training the DAAE model:

Table 2: Extra hyperparameters for DAAE model; hyperparameters from table 1 remain unchanged.

Parameter	Symbol	Value
Word Drop Probability	p_{drop}	0.1
Word Blank Probability	p_{blank}	0.1
Word Replace Probability	p_{replace}	0.1
Max Shuffle Distance	d_{shuffle}	3
Adversarial Loss Weight	λ	0.5

5 Results

We now obtain recommendations for a handful of query titles to qualitatively evaluate the performance of the trained autoencoder and denoising adversarial autoencoder (DAAE) models for the task of paper recommendation. For each query title, we present the three nearest neighbors (measured by eudclidean distance) that were found in the latent spaces of the autoencoder and DAAE. The query titles were chosen from an unseen set of 20,000 research paper titles. The remaining 180,640 research papers found in the arXiv database were included in the reference database.

In table 3, we consider three examples where the DAAE provided more relevant recommendations than the autoencoder.

Table 3: Examples of nearest neighbors in the latent Euclidean space of AAE and DAAE that demonstrate strong performance by the DAAE.

Source	**"Towards Optimal Neural Networks: The Role of Sample Splitting in Hyperparameter Selection"**
3-NN by AAE	"Towards Applicable Reinforcement Learning: Improving the Generalization and Sample Efficiency with Policy Ensemble" "Using Meta Reinforcement Learning to Bridge the Gap between Simulation and Experiment in Energy Demand Response"

	“Revisiting Game Representations: The Hidden Costs of Efficiency in Sequential Decision-Making Algorithms”
3-NN by DAAE	“Socially Supervised Representation Learning: The Role of Subjectivity in Learning Efficient Representations” “Towards Understanding Neural Collapse: The Effects of Batch Normalization and Weight Decay” “Efficient Adversarial Contrastive Learning via Robustness-Aware Coreset Selection”
Source	**“Combining Model-Predictive Control and Predictive Reinforcement Learning for Stable Quadrupedal Robot Locomotion”**
3-NN by AAE	“A Privacy-Preserving Data Storage and Service Framework Based on Deep Learning and Blockchain for Construction Workers’ Wearable IoT Sensors” “Integrating Tabu Search and VLSN Search to Develop Enhanced Algorithms: A Case Study Using Bipartite Boolean Quadratic Programs” “Integrating Kinematics and Environment Context into Deep Inverse Reinforcement Learning for Predicting Off-Road Vehicle Trajectories”
3-NN by DAAE	“Integrating Kinematics and Environment Context into Deep Inverse Reinforcement Learning for Predicting Off-Road Vehicle Trajectories” “Combining Reinforcement Learning with Model Predictive Control for On-Ramp Merging” “Enforcing the Consensus Between Trajectory Optimization and Policy Learning for Precise Robot Control”
Source	**“Explaining and Visualizing Black-Box Models Through Counterfactual Paths”**
3-NN by AAE	“Seeing Seeds Beyond Weeds: Green Teaming Generative AI for Beneficial Uses” “OptimalFlow: Optimal-Transport Approach to Flow Cytometry Gating and Population Matching” “RoBERTuito: A Pre-Trained Language Model for Social Media Text in Spanis”
3-NN by DAAE	“Explaining Black-Box Models Through Counterfactuals” “Interpreting and Improving Deep-Learning Models With Reality Checks” “Empowering the Trustworthiness of ML-Based Critical Systems Through Engineering Activities”

Reviewing the examples in table 3, we notice a strong thematic coherence between the query titles and the papers recomended by the DAAE. In the first example, the query title and the three recommendations from the DAAE latent space fall under the heme of “neural network optimization and learning“. The second recommended paper on neural collapse is especially relevant as it describes specific hyperparameters to learn optimal neural networks. The first and third papers deal with learning generalizable representations, which is the goal of network optimization. In contrast, two of the recommendations for the autoencoder went off topic, discussing reinforcement learning which is a separate topic from supervised learning studied by the query paper. The third recommendation from the autoencoder may seem related to the query paper due to the presence of the word “efficiency”, however the context of sequential decision making is unrelated to the query paper’s context. Similarly, the second and third examples demoonstrate how the query title and recommendations from DAAE latent space focus on themes of “motion planning and control” and “explainability for black box models” respectively.

In table 4, we consider three examples where the autoencoder latent space provided recommendations that were of similar quality to the recommendations from DAAE latent spaces, and in some cases of superior quality.

Table 4: Examples of nearest neighbors in the latent Euclidean space of AAE and DAAE that demonstrate strong performance by the autoencoder.

Source	**"A Secure Aggregation for Federated Learning on Long-Tailed Data"**
3-NN by AAE	"DPFormer: Learning Differentially Private Transformer on Long-Tailed Data" "A Call for More Rigor in Unsupervised Cross-Lingual Learning" "Federated Learning with Server Learning: Enhancing Performance for Non-IID Data"
3-NN by DAAE	"RingFed: Reducing Communication Costs in Federated Learning on Non-IID Data" "A Secure Federated Learning Framework for Residential Short-Term Load Forecasting" "Personalized Cross-Silo Federated Learning on Non-IID Data"
Source	**"Neurosymbolic AI for Reasoning on Biomedical Knowledge Graphs"**
3-NN by AAE	"Optimization of Retrieval Algorithms on Large Scale Knowledge Graphs" "Neurosymbolic AI for Reasoning on Graph Structures: A Survey" "Neurosymbolic AI for Situated Language Understanding"
3-NN by DAAE	"Swift Markov Logic for Probabilistic Reasoning on Knowledge Graphs" "EXPERT: Public Benchmarks for Dynamic Heterogeneous Academic Graphs" "Pretrained Transformers for Simple Question Answering over Knowledge Graphs"
Source	**"HeroLT: Benchmarking Heterogeneous Long-Tailed Learning"**
3-NN by AAE	"FedHQL: Federated Heterogeneous Q-Learning" "Towards Effective Collaborative Learning in Long-Tailed Recognition" "Towards Federated Long-Tailed Learning"
3-NN by DAAE	"POPGym: Benchmarking Partially Observable Reinforcement Learning" "FedHQL: Federated Heterogeneous Q-Learning" "OntoZSL: Ontology-Enhanced Zero-Shot Learning"

The second example of table 4 demonstrates a case where the autoencoder provides may provide relevant recommendations by honing recommendations to a research sub-field, which is "neurosymbolic AI" in this example. Meanwhile the decoder attempts to generalize its recommendations the larger theme of knowledge graphs, which may be to broad of a research theme for researchers interested in neurosymbolic AI.

Lastly, we highlight in table 5 cases where recommendations from neither the autoencoder latent space nor the DAAE latent space were particularly relevant to the query title.

Table 5: Examples of nearest neighbors in the latent Euclidean space of AAE and DAAE where both models performed poorly.

Source	**"Neural Stream Functions"**
3-NN by AAE	"Deep Information Propagation" "Deep Submodular Functions" "Probabilistic Neural Programs"
3-NN by DAAE	"Deep Neural Maps" "Neural Power Units" "Neural Integral Equations"
Source	**"MESOB: Balancing Equilibria & Social Optimality"**
3-NN by AAE	"LEAF: Latent Exploration Along the Frontier" "FlexiAST: Flexibility is What AST Needs"

	“MR: Evaluating NER Recall on Tough Mentions”
3-NN by DAAE	“FlexiAST: Flexibility is What AST Needs”
	“Boldly Going Where No Prover Has Gone Before”
	“STARDATA: A StarCraft AI Research Dataset”

Observe that both the autoencoder and DAAE latent spaces fail to find similar titles for the short query title “Neural Stream Functions”. Similar failures occur when finding recommendations for papers that introduce a novel technique that does not appear elsewhere in the reference database. We elaborate more upon the failure modes of our method in Section 7.

6 Discussion

Through our qualitative evaluation of the latent space constructed by the autoencoder and the DAAE, we found that the DAAE’s latent space features stronger thematic “clusters” than the autoencoder’s latent space. The autoencoder’s latent space instead relied mostly on shared keywords to group papers, acting as a “soft” keyword based search. Both properties of the DAAE and autoencoder latent space may be beneficial to recommendations at different stages of the literature search. Indeed, the generalizability of DAAE embeddings may be useful for providing recommendations under a bigger theme in the early stages of the literature review, while the autoencoder embeddings may be more helpful in providing focused recommendations once the researcher has chosen a specific sub-field of research.

7 Limitations

A notable limitation of our approach is the autoencoder reconstruction loss which aims to preserve structure of its inputs (with length being a key part of title structure). As a result, there was a degradation of recommendation quality for very short sequences which was observed in Section 5 for both autoencoders and DAAEs. This limitation of our approach may be mitigated by addressing the imbalance of title lenghts in figure 2, ensuring a roughly uniform distribution of title lengths in the reference database. Another limitation which is more difficult to address is the fixed vocabulary of our autoencoder, resulting in low-quality recommendations when the query title contains a keyword that is not found in the autoencoder vocabulary as in the second example in table 5.

8 Ethical Considerations

In this section we address concerns about the use of recommendation systems for academic work. Consider the hypothetical example of Peter, a computer science researcher who publishes his work in IEEE Xplore. Suppose further that our system is used in the IEEE Xplore database, with recommendations being provided to every user (i.e. there is no opt-out). In this scenario, Peter, who publishes his work in a niche field of computer science, may claim that the system is biased towards recommending papers from more active research fields. As such, he believes that his work receives fewer views (and citations), which influences measures (e.g. h-index) of his productivity as a researcher.

The possibility for bias in the data used to train the recommendation system lends credence to Peter’s claim. For example, researchers aware of a recommendation system may be incentivized to include certain keywords in titles to boost the visibility of their work. This can cause a recommendation system trained on this data to “over-recommend” certain research papers. As this source of bias is more difficult to control, eliminating bias from data is a non-trivial endeavor. Therefore, we aim to make our use use of data transparent by using publicly available data that has been published voluntarily and documenting our data processing steps.

9 Conclusion

We propose a recommendation for research papers using embeddings of research paper titles learned by an autoencoder and denoising adversarial autoencoder (DAAE). We perform a qualitative evaluation of the $k-$nearest neighbors recommendations in the DAAE and autoencoder latent space, observing that the latent space of the DAAE captures thematic clusters more clearly. We believe this property of the DAAE latent space can facilitate relevant recommendations, especially for junior researchers who wish to find similar works falling under a broader theme. However, in some cases we observed that the autoencoder latent space provided more pointed recommendations (see Table 5) and remarked on the similarity of these recommendations to results from a traditional keyword search. In conclusion, we determined that while the DAAE represents an improvement over the baseline autoencoder, the recommendations are primarily helpful for junior researchers who have not established a focused research sub-field for their searches.

References

[1] X. Bai, M. Wang, I. Lee, Z. Yang, X. Kong, and F. Xia, "Scientific paper recommendation: A survey," *IEEE Access*, vol. 7, pp. 9324–9339, 2019.

[2] C. K. Kreutz and R. Schenkel, "Scientific paper recommendation systems: a literature review of recent publications," *Int. J. Digit. Libr.*, vol. 23, pp. 335–369, Oct. 2022.

[3] B. Oshri and N. Khandwala, "There and back again: Autoencoders for textual reconstruction," 2015.

[4] T. Shen, J. Mueller, R. Barzilay, and T. Jaakkola, "Educating text autoencoders: Latent representation guidance via denoising," 2020.

[5] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, "Recommender systems," *Physics Reports*, vol. 519, no. 1, pp. 1–49, 2012. Recommender Systems.

[6] M. J. Pazzani and D. Billsus, *Content-Based Recommendation Systems*, pp. 325–341. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[7] S.-j. Lin, G. Lee, and S.-L. Peng, "Academic article recommendation by considering the research field trajectory," in *Intelligent Computing Paradigm and Cutting-edge Technologies* (M. N. Favorskaya, S.-L. Peng, M. Simic, B. Alhadidi, and S. Pal, eds.), (Cham), pp. 447–454, Springer International Publishing, 2021.

[8] S. Ahmad and M. T. Afzal, "Combining metadata and co-citations for recommending related papers," *TURK. J. OF ELECTR. ENG. COMPUT. SCI.*, vol. 28, pp. 1519–1534, May 2020.

[9] A. M. Nair, O. Benny, and J. George, "Content based scientific article recommendation system using deep learning technique," in *Inventive Systems and Control* (V. Suma, J. I.-Z. Chen, Z. Baig, and H. Wang, eds.), (Singapore), pp. 965–977, Springer Singapore, 2021.

[10] K. Sugiyama and M.-Y. Kan, "Exploiting potential citation papers in scholarly paper recommendation," in *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '13, (New York, NY, USA), p. 153–162, Association for Computing Machinery, 2013.

[11] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*, pp. 1–35, Boston, MA: Springer US, 2011.

[12] X. Zhao, H. Kang, T. Feng, C. Meng, and Z. Nie, "A hybrid model based on lfm and bigru toward research paper recommendation," *IEEE Access*, vol. 8, pp. 188628–188640, 2020.

[13] P. Subathra and P. N. Kumar, "Recommending research article based on user queries using latent dirichlet allocation," in *Soft Computing and Signal Processing* (V. S. Reddy, V. K. Prasad, J. Wang, and K. T. V. Reddy, eds.), (Singapore), pp. 163–175, Springer Singapore, 2020.

[14] G. Guo, B. Chen, X. Zhang, Z. Liu, Z. Dong, and X. He, "Leveraging title-abstract attentive semantics for paper recommendation," *Proc. Conf. AAAI Artif. Intell.*, vol. 34, pp. 67–74, Apr. 2020.

[15] D. Bank, N. Koenigstein, and R. Giryes, *Autoencoders*, pp. 353–374. Cham: Springer International Publishing, 2023.

[16] A. Collins and J. Beel, "Document embeddings vs. keyphrases vs. terms for recommender systems: A large-scale online evaluation," in *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pp. 130–133, 2019.

[17] arXiv.org submitters, "arxiv dataset," 2023.