bad pun: Balanced Article Discovery through Playful User Nudging

Mara-Jean Krupa **Joshua Oehms** **Vera Wesselkamp**

Abstract

To reduce the risk that selective exposure to news through recommender systems may lead to societal polarization, the research community introduced metrics such as diversity and coverage. However, even balanced systems fail to provide transparency. We hence introduce badpun, a recommender system that educates users about its inner working, while giving them the autonomy over their own news consumption. We achieve this through combining an online learning-based recommender system, visualization of its recommendations via LRP and attention mechanisms, and the presentation of alternate feeds through clustering of other users' recommendations. To evaluate the quality of the clusters, we additionally introduce three sanity checks to determine suitable hyperparameters for dimensionality reduction and clustering. We show that badpun achieves state-of-the-art recommendation accuracy and outputs distinguishable feed recommendations for unseen headlines.

1 Introduction

In the increasingly fast moving digital news economy, modern recommender systems are often considered the only viable solution to cut through the masses of fresh articles and deliver a relevant newsfeed to users without overloading them. In order to keep up with the rapid aging of articles, providers of recommendation solutions such as *Yusp* re-train their models every few minutes to deliver up-to-date recommendations (Vas, 2022). As an alternative to resource intensive re-trainings, online learning, which continuously updates models in small incremental steps, has emerged as a field of research (Ermis et al., 2022; Hadsell et al., 2020).

However, users tend to prefer articles that adhere to their own worldviews. As a consequence, strongly personalized recommendations foster intellectual isolation, an effect called *selective exposure* (Knudsen, 2023). While research on online filter bubbles is still inconclusive, and generally shows that they are much less common than generally considered (Ross Arguedas et al., 2022), metrics like diversity, novelty, serendipity, and coverage (Raza and Ding, 2022) are all active fields of research in the recommender community, striving to combat potential risks of isolation. However, even systems that integrate diverse articles into a user's feed still fail to demonstrate such an improvement to the user. In fact, a study in 2018 has shown that about 90% of all Germans fear that the availability of personalized news can increase societal polarization, and inhibits people from being comprehensively informed (Oertel et al., 2023). This perceived threat is representative of the lack of understanding of effects and risks of recommender systems. As news media are an integral part of a well functioning democracy, and trust in its institutions is essential, we believe it to be of great importance to both educate users about recommendation systems, and give them autonomy in shaping their news consumption.

We thus build an online learning recommender system using adapters, which demonstrates the effect of personalization to the user by (i) providing explanations for the article recommendations, and (ii) spatially visualizing the system's perception of the user in relation to other users. In addition, the system also offers the display of representative alternative user feeds and their interpretation via a clustering, ultimately nudging the user towards a balanced news consumption.

In order to derive both the representative alternatives and the visualization, we employ dimensionality reduction on the high dimensional user embeddings derived from the recommender system. To perform unsupervised clustering with the goal of interpretable clusters in mind, we strike a delicate balance between proximity preservation and sensible non-linear dimensionality reduction. We closely evaluate different parameters by perform-

ing a hyperparameter search and conducting three sanity-checks that match our criteria.

Ultimately, our contributions are thus:

- We design a recommender system with capabilities for real-time adaption to user preferences.
- We develop three invariants to test the spatial preservation of dimensionality reduction and interpretability of feed clustering, and provide both explanations and interpretations for the displayed feeds using accurate visual representations.
- We build a system which combines these two elements to provide users with a personalized news consumption, while at the same time educating them both about the effect of this personalization, as well as providing them with options to balance their news consumption based on other users profiles.

2 Related Work

Badpun combines several components to achieve transparency and educate users about its effect. We present work that serves as a basis for the individual components.

2.1 Explainability

In recent years, explainability of machine learning has become an important research focus (Gunning et al., 2019), pushing the field of recommender systems to provide explanations to users. For example, Li et al. (2021) suggest using **personalized transformer** models for explanations. However, different from us, they use transformer models to provide explanations in continuous text for recommended items. Since our "items" already contain text, we base the explanation on the existing words.

Model-agnostic methods such as LIME and SHAP can be used to create explanations for Natural Language Processing (NLP) classification problems (Naylor et al., 2021). However, since we have access to the code and weights of our models, built-in explainability methods, such as Layer-wise Relevance Propagation (LRP), that can be executed at inference time (in parallel to the prediction) are better suited to our system (and for real-time applications in general).

Attention-based approaches use the classification token's attention distribution on the last layer. Although attention scores have not been shown to equate with interpretability, they are often used to visualize and interpret transformer models (Abnar and Zuidema, 2020).

LRP-based approaches propagate relevancy scores through the entire network. Therefore, Bach et al. (2015) propose LRP for image classification tasks. Eventually, the scores can be used to access the impact of certain tokens to the final classification (Arras et al., 2016). More recently, Chefer et al. (2021) used a combination of LRP and attention distributions to generate explanations for transformer predictions.

2.2 Online Learning

Online learning (or often referred to as "continual learning") is required to continuously train the model on the user's current interests. One could simply fine-tune all parameters constantly in the same style as in pre-training. However, in this setup, catastrophic forgetting can occur quickly, and the computational costs are high. Therefore, Ermis et al. (2022) propose using adapters for continual learning with transformer models.

Adapters Many NLP downstream tasks base their performance in transfer learning. However, the dedicated fine-tuning of pre-trained transformer models requires large computational resources and, after training is completed, the corresponding storage space. When using adapter layers, these are added to the existing pre-trained transformer and only these adapters are trained (Pfeiffer et al., 2020). As a result, the number of trained parameters can be drastically reduced and thus the computing effort and required hard disk memory.

2.3 Clustering and Interpretability

Providing cluster interpretability using automated cluster descriptions is an active field of research with a large diversity of approaches. De Cao et al. (2020) suggest a multi-objective optimization during the clustering process, enriched by predefined user tags. During the optimization, tag agreement within clusters is maximized while the diameters of the user clusters are minimized. The tags subsequently serve as cluster descriptions. However, as we do not have any predefined user tags, this approach does not constitute a viable solution for our user embedding clustering. Davidson et al. (2022) propose an approach closely coupled to *Concept Theory* in cognitive psychology. They argue that using features as a basis of explanation

is not a suitable approach when working with high-dimensional, hardly interpretable features. Therefore, they instead use cluster exemplars as a means of explanation, as this enables the user to leverage her preexisting knowledge and expedites the internalization of information. This idea is highly applicable to our cluster interpretation endeavors, as the user embedding features are high-dimensional and lack in interpretability. Finally, Rashed et al. (2021) make use of wordclouds to describe the communities resulting from their clustering endeavors. We use LRP and attention values to access information about the influence of specific words on the click prediction of an article, and provide additional cluster interpretability through wordclouds.

3 System Design

We design a system which (i) personalizes user news recommendation while continuously integrating the user's feedback on liked articles, (ii) explains the effect of her feedback on the recommendations to the user, and (iii) provides alternative news feeds for a playful discovery of diverging interests and opinions. We accordingly divide the system into three components. Figure 1 shows the system architecture.

The first component is the online learning recommendation system (described in detail in Section 4), which works in the following way:

1. Our **pre-trained click predictor** is initialized from a BERT[1] model and fine-tuned on the MIND[2] dataset.

2. The **personalized click predictor** is retrieved from the pre-trained model by adding adapters as a last layer. The click predictor receives article headlines as an input and outputs scores for each of these headlines, depending on how closely they fit the user's preference.

3. The scores get passed to the **ranking module**, which introduces diversity in the recommended headlines by filtering close duplicates using the *rouge score*. Recommended headlines are shown in the user interface, ordered by recommendation rank.

4. In the interface, the headlines are shown one by one, allowing a user to either click or skip the article. Her actions are sent back through the personalized click predictor in a backwards pass to **update her embedding**.

The second component is the explanation component, which provides users with a visualization of the decisions made by the recommender, and is described closer in Section 5. It consists of:

(a) The **keyword selector** which, simultaneously to a recommendation forward pass, generates values for each word of a headline, indicating its influence on the recommender's decision. We provide two ways to generate the values, via the last-layer-attentions and via the explanation generated by LRP.

(b) The **wordcloud module**, which preprocesses the results of the keyword selector and visualizes the words in a wordcloud.

Finally, the last component is the (I) **Clustering Module**, which performs dimensionality reduction and clusters the user embeddings of the pre-trained model to extract representative alternative feeds. We describe this component in Section 6.

To solve the problem of a cold start for a recommendation system, on start we provide the user with three significantly different profiles. We set the initial embedding to that of the profile chosen by the user.

Interface: We built a prototype of the system using the python library `streamlit`. Screenshots of the interface can be found in Figure 2.

4 Continuous Integration of User Feedback

We pre-train the personalized click predictor on the MIND dataset (Wu et al., 2020), which contains news headlines and corresponding users who have either clicked or not clicked these article headlines. At this stage, we only train the classifier-head, the user embedding matrix (and optional user embedding projection matrix), and the entire batchnorm weights. We compare three different approaches to build a recommender system that both is applicable to online learning, and provides user embeddings that can be used for clustering:

(1) Using the **average article embeddings** of articles clicked by a user as her embedding, obtained from a Sentence-BERT[3], we calculate the dot products of normalized candidate article embeddings

[1] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[2] https://msnews.github.io/

[3] (Reimers and Gurevych, 2019)

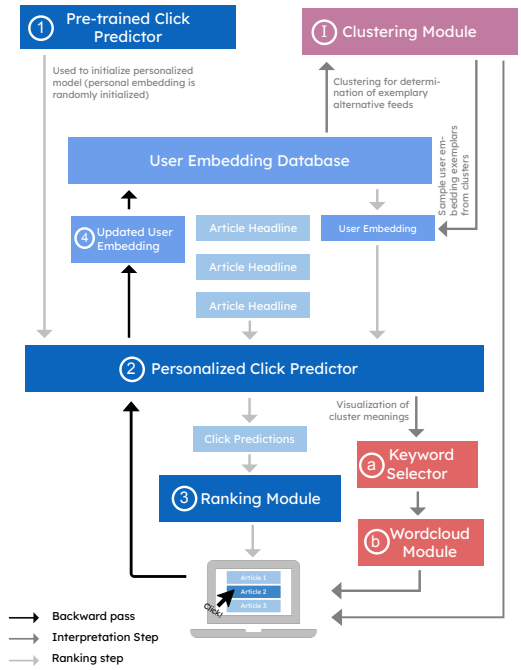

Figure 1: The *badpun*-system. Its three main components are the recommendation system (blue), the clustering module used for constructing alternative user profiles (pink) and the interpretation module (red).

and user embeddings. We define a threshold for positive and negative predictions.

(2) User embeddings are trained to minimize the **dot product** for clicked news-sentence-embedding (and maximize for non-clicked news). Additionally, the news-embedding is personalized using user-specific attention layers at the final pooling step, as proposed by Wu et al. (2019). Finally, a classifier-head learns to transform dot-product values into probabilities.

(3) **Personalized adapters**: The users are considered each as a "downstream task". These tasks are trained using $(IA)^3$-like adapters on the last transformer layer (Liu et al., 2022). In contrast to the original $(IA)^3$ implementation, we only scale key and value vectors. Finally, the user embeddings are the adapter's learned values.

4.1 Pre-training

As we face the issue of an imbalanced dataset, we use undersampling to artificially balance the data. We balance the negative/positive ratio globally and locally per user, and filter the data to only contain users with at least ten click actions. We use contrastive loss to push embeddings of different users apart and pull similar embeddings close together.

We evaluate two different training objectives:

(i) Training with a single clicked/not clicked label for all users and their presented headlines, which is very fast as we can train all user decisions on a certain headline at once, but yields poor results compared to state-of-the-art.

(ii) Training with negative sampling ($k = 4$) as proposed by Wu et al. (2019). At each training step, pick one positive headline and four negative headlines for a certain user, then train the model to predict which of them is the positive one. This approach achieves state-of-the-art results, but is significantly slower than the first approach.

While the personalized adapters approach performs slightly better than the learned dot product approach, training on our initial training objective still cannot match the performance of the (untrained) average-embedding approach. However, as depicted in Table 1, introducing the negative sampling objective, the results are close to the best score reported by the MIND authors. Finally, we choose the adapter approach as the performance is close to state-of-the-art and as it provides the best solution for a personalized neural network, which can also be easily "de-personalized" again.

To fulfill our additional objective of providing good and fast-processable user representations for clustering, we compare the training of user embeddings of different dimensionality. The dimensionality provided by the pre-trained BERT (Devlin et al., 2019) is 384 and thus our adapter-vector is twice as big (as we have to scale key **and** value vectors). We notice no significant change in performance, when reducing the dimension from 768 to 32 using a projection matrix similar to the ALBERT embedding projection (Lan et al., 2019). Reducing the dimensionality prevents overfitting. However, for the 32 dim approach, we find that calculating contrastive loss directly on the reduced user embeddings decreases accuracy and calculating contrastive loss after projecting the reduced user embeddings (to 768 dim) leads to bad clustering results. Finally, we choose a dimensionality of 128 and calculate contrastive loss before projection for a good trade-off between clustering speed and performance.

Since we evaluate our system on 300 unseen newly scraped online news-headlines[4] getting good results, we can assume a well generalized model.

[4]Source: Foxnews, LA-times, CNN, NY Times, USA Today, ABC News

Figure 2: The badpun-interface. The first image shows the cold-start interface, where users choose an initial preference. The second image shows the personalized recommendation including visualization and explanation. The third image shows the representative alternative feeds.

	Positive / negative	Negative sampling
Wu et al. (2020)	-	*68.23%*
Average embedding	57.57% (Not trained)	
Dot product	53.51%	-
Adapters	54.58%	63.07%
Adapters (OL)	-	**73.00%**

Table 1: The performance of each version (Area under Curve) of the click predictor compared to the best model evaluated by the MIND authors. Average embeddings are not trained, the value serves as a baseline. Adapters (OL) is the adapter model trained for 20 online learning steps with randomly initialized user embeddings.

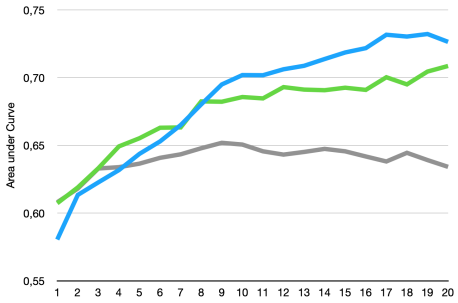

Figure 3: Online learning performance on unseen articles. Blue: all negative samples are available. Green: ten negative samples. Gray: two negative samples. We used about 400 users for these experiments. The number of training steps is depicted on the x-axis (equal to the number of positive user feedback).

4.2 Online Learning

For online learning, we only train the user embeddings, and hence freeze the entire model weights (including the optional user embedding projection matrix) except the actual user embedding matrix / adapter value matrix.

As a training objective, we use negative sampling ($k = 4$). We compare the impact of several aspects such as batch size, learning rate, and the minimum number of negative and positive samples. For the experiments, we reset all user embedding weights to random values. We found that (i) there is no significant difference between using all negative samples accumulated over time and using exclusively recent negative samples, (ii) there is no significant difference using different batch sizes (and a learning rate of 0.0025 performs best in our case), (iii) we need at least five update steps (with five different positive samples) to achieve comparable performance, and (iv) we need at least ten negative samples to improve over time (visualized in Figure 3)

We collect all negative samples over time and update the user embedding weights at each positive or negative feedback action using a batch size of one and a learning rate of 0.0025.

5 Explaining Recommendations

In a first approach, we use the **attention** distribution of the [CLS] token on the last layer and compare the personalized (user-specific adapters enabled) against the non-personalized attentions (user-specific adapters disabled = the pure pre-trained model). We merge subword-tokens into words and calculate the ratio between the two attention scores for each word.

In a second approach, we use **LRP** and follow Chefer et al. (2021) adapting their code[5] for our purpose. We merge subword-tokens and use the scaled LRP relevancy values for each word. The scores range between 1.0 for the most important word and 0.0 for the least important word.

We post-process the results to retrieve the word-clouds. The details of the processing can be found in Appendix A.

[5] https://github.com/hila-chefer/Transformer-Explainability

6 Retrieving Cluster Representations

The clustering module comprises three main steps: a potential preprocessing of the user embeddings, the dimensionality reduction of the embeddings to three-dimensional space, and finally the clustering of the embeddings.

6.1 Data Preprocessing

Reducing the impact of data skewness, data preprocessing is a viable means for enhancing model performance (Kuhn and Johnson, 2013). As the benefit of different types of preprocessings can only be determined manually, we treat the type of preprocessing as an additional hyperparameter for the dimensionality reduction step. We test three different data preprocessing approaches for our user embeddings:

Embedding normalization converts embedding vectors into unit vectors, thereby projecting them onto a unit sphere, *Feature standardization* transforms data to have unit variance and zero mean, and finally *No data preprocessing*.

6.2 Dimensionality Reduction

To visualize the resulting clustering, the user embeddings' dimensionality has to be reduced from the 128-dimensional space to the three-dimensional space. We consider two transformations as options for the dimensionality reduction: Principal Component Analysis (PCA) and Uniform Manifold Approximation and Projection (UMAP). As UMAP is known to well preserve both the local and global structure of the data when projecting it down to the lower dimension (McInnes et al., 2020), and first tests reveal that PCA is not capable of capturing the complexity of our user embeddings in the low-dimensional space, we choose UMAP for dimensionality reduction.

6.2.1 Hyperparameter Search

As UMAP requires the specification of four hyperparameters (McInnes et al., 2020), we conduct a hyperparameter search. Table 2 shows the explored values for each parameter. We perform one UMAP transformation for each possible combination of values sampled from the hyperparameter ranges, additionally applying each of the three embedding preprocessing options.

Figure 4 shows a comparison of exemplary results returned from UMAP transformations using different hyperparameters. It shows that the *Minimum Distance* parameter does not influence the structure of the data in the lower-dimensional space but rather the tightness of the resulting clustering. As it is our objective to later cluster the embeddings, we choose the low parameter value of 0.4.

Parameter	Explored	Chosen
No. of components	3	3
Min. distance	[0,10]	0.4
Nearest neighbours	[2,50]	
Metric	euclidean, manhattan, cosine	cosine

Table 2: Parameter ranges for UMAP

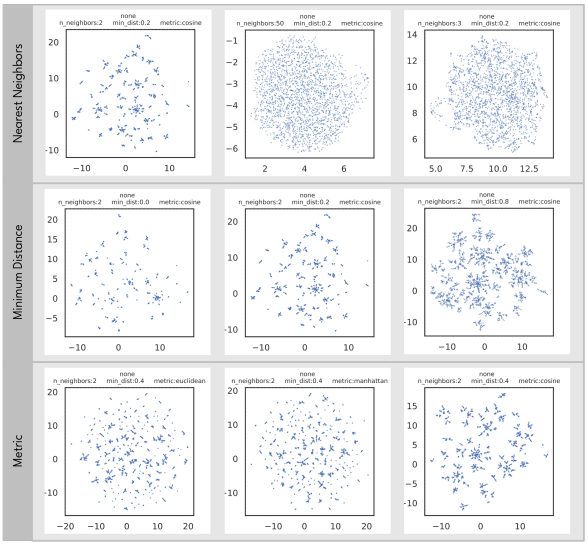

Figure 4: Exemplary results outputted by the UMAP dimensionality reduction of our (non-preprocessed) user embeddings with different hyperparameters.

For *Metric*, the data preprocessing has an influence on the choice of parameter values. When using *no preprocessing* or *feature standardization*, the cosine metric results in the clearest clustering. When using *embedding normalization* however, the metric makes no difference. As *embedding normalization* projects the embeddings onto a unit sphere, all information not related to the phase of the vector is discarded, focusing the distance calculation on the angular information even when not using the cosine metric. Since we also use cosine similarity in our click predictor, we choose it as our metric.

The *Nearest Neighbor* parameter determines the size of the local neighborhood taken into consideration when constructing a graph representation of the numerical data, and hence strongly influences the structure of the embeddings in the low-

dimensional space. While *Nearest Neighbor* $= 2$ yields a well-distinguishable clustering of the user embeddings, larger values yield an unstructured appearance (blob), showing no clear community structure. To decide between the two options, we perform sanity checks on the resulting clustering before choosing the precise parameter value.

6.2.2 Sanity Checks

Due to the non-linearity of the UMAP transformation as well as the lack of knowledge about the expected shape and number of the embedding clusters in the lower dimensional space, we derive sanity checks to validate the results of different hyperparameters and determine the value of the *Nearest Neighbour* parameter.

We reason that the dot products of an article embedding with two user embeddings that are close to each other in the high-dimensional space shall yield a similar click prediction for both users given an arbitrary article headline. Consequently, we infer two fundamental assumptions for the sanity checks: (i) Two user embeddings that are close to each other in the high-dimensional space lie within the same cluster. (ii) Two user embeddings that are distant from one another in high-dimensional space lie in separate clusters.

Note that in unstructured results (UMAP transformation with *Nearest Neighbor* > 2), *clusters* refer to the close neighborhoods of the respective embeddings. These neighborhoods can be imagined as a grid-like structure.

Based on the assumptions, we contrive three sanity checks for the hyperparameter configurations:

1. The nearest neighbors of an embedding in high-dimensional space should lie within the same cluster as this embedding in low-dimensional space.

2. Two embeddings that lie far apart in high-dimensional space should lie within separate clusters in low-dimensional space.

3. A group of embeddings that lie within the same cluster should produce similar feeds.

The first two sanity checks validate the successful proximity preservation amongst a group of user embeddings throughout the transformation. The last sanity check is used to prove our initial assumptions. We do not test for embeddings from different clusters to have dissimilar feeds, as we allow our model to have a redundant community structure.

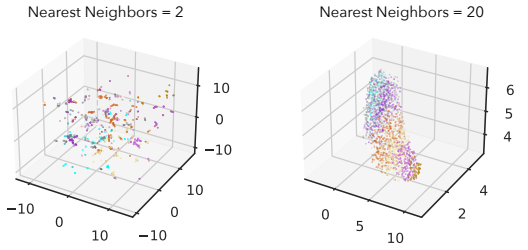

Figure 5: Results of sanity check one for two different *Nearest Neighbor* parameter values

For (1.), we perform k-means clustering for eight clusters in the high-dimensional space and display the results of the clustering in the low-dimensional space. A visual analysis then yields the quality of proximity preservation. For (2.), we randomly sample a user embedding and determine the user embedding furthest away from the sampled embedding in high-dimensional space. We then inspect if these two embeddings are also distant from one another in low-dimensional space. For (3), we sample user embeddings from our embedding database and manually assign one out of five labels to the respective users, labeling their feeds' content as sports-oriented, celebrity-oriented, food-oriented, domestic-politics-oriented, or international-politics-oriented. We then display the user-theme-groups in the lower-dimensional space.

When performing the sanity checks, it becomes apparent that the proximity preservation does not work well for a *Nearest Neighbor* $= 2$ (see Figure 5). While this hyperparameter setup keeps two distant embeddings far apart from one another in the lower-dimensional space, it cannot retain the proximity of more than one nearest neighbor. Furthermore, as can be seen in Figure 6, the clusters for *Nearest Neighbor* $= 2$ contain several different feed themes at once. Hence, users that lie within the same cluster do not necessarily have similar feeds. Given this analysis, *Nearest Neighbors* $= 2$ is not a viable parameter choice.

As can be seen in Figure 5, instead, for *Nearest Neighbors* > 2 the proximity preservation achieves high-quality results when using *no preprocessing* or *feature standardization*.

Finally, the third sanity check validates our initial assumption that user embeddings that lie close to each other produce similar feeds. As can be seen in Figure 6, each theme is assigned a region of its own in the lower-dimensional space. Notably, the

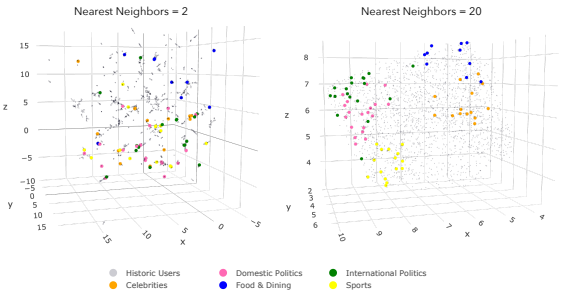

Figure 6: Results of sanity check four for two different *Nearest Neighbor* parameter values

recommender system even manages to differentiate fine-grained themes like international and domestic politics.

As there is no quality difference in the results of the sanity checks performed with feature standardization and no preprocessing, we opt to discard the preprocessing step as this reduces the computational complexity of our application. As all metrics perform equally well for the chosen configuration, the Cosine metric seems to be the natural choice given that we also use Cosine similarity within the click predictor. The final parameters can be found in Table 2.

6.3 Clustering

Because of the great quality in proximity preservation of our hyperparameter setup, we opt for k-means as a clustering algorithm due to its simplicity. Notably, we cluster in low-dimensional space due to the previously observed locality of the feed themes. We manually test different numbers of clusters and inspect the feeds of the resulting cluster representatives to test whether they match the feeds of users sampled from the same cluster. This manual inspection yields a preferable number of clusters equal to 45. The resulting clustering appears to be redundant. As the cluster boundaries are drawn randomly by the k-means algorithm, not every cluster assignment fits perfectly for every user. Nonetheless, the results are highly satisfying for presenting alternative feeds to an individual user.

7 Discussion

According to Eskens (2022), by the rule of law, media should be unrestricted, making regulation on how users should select their news a very sensitive topic. At the same time, many people do not consume news from media houses directly, and instead rely on "very large online platforms", such as Google News, Microsoft News (where the MIND dataset originates from), or even social media such as YouTube or Facebook. These platforms do fall under the regulation of an EU regulation proposal that demands both transparency on the parameters used for regulation, and giving users the possibility to use a non-personalized system. Our system falls in line with these demands.

There are issues with recommender systems that remain unaddressed in our proposal. A major problem raised by all personalization is that of privacy. In order to personalize to the user's preferences, all the user's history must be tracked. Sharing the user's preferences with the news outlet can be solved by running the personalized model on the user's device, thus keeping all information in the user's hands. This presupposes a model that is small enough to run on mobile devices, and would put more load on the network. Secondly, our visualization of users in space automatically publishes some information about the historic users used to train the model. In order to avoid the reconstruction of user information via deanonymization attacks, techniques such as differential privacy (Dwork, 2006) can be used to aggregate data.

Future Work While the system already includes a measure of diversity through the filtering of similar recommendations, we neglected the implementation of other criteria, such as novelty, serendipity, or coverage, that would be necessary to handle a live stream of articles. We additionally aim to allow a user to proactively shape their recommendations by moving their feed in the direction of a chosen alternative feed. Moreover, automatic cluster titling via keywords would give clusters a clearer meaning and improve the user experience when interacting with the alternative feeds. Lastly, it would be of interest to integrate the state-of-the-art work on cluster interpretability, as described in Section 2.

8 Conclusion

We presented badpun, an online learning news recommender that provides transparency and educates users about the effect of personalized recommendations. We constructed a recommender that generalizes well to unseen headlines and developed three sanity checks demonstrating that our clusters clearly separate the space into localizable and distinguishable user themes.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2016. Explaining predictions of non-linear classifiers in NLP. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 1–7, Berlin, Germany. Association for Computational Linguistics.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140.

Hila Chefer, Shir Gur, and Lior Wolf. 2021. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 782–791.

Ian Davidson, Michael Livanos, Antoine Gourru, Peter Walker, Julien Velcin, and SS Ravi. 2022. Explainable clustering via exemplars: Complexity and efficient approximation algorithms. *arXiv preprint arXiv:2209.09670*.

Nicola De Cao, Michael Sejr Schlichtkrull, Wilker Aziz, and Ivan Titov. 2020. How do decisions emerge across layers in neural models? interpretation with differentiable masking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3243–3255, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Cynthia Dwork. 2006. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer.

Beyza Ermis, Giovanni Zappella, Martin Wistuba, Aditya Rawal, and Cedric Archambeau. 2022. Memory efficient continual learning with transformers. *Advances in Neural Information Processing Systems*, 35:10629–10642.

Sarah Eskens. 2022. Regulating news recommender systems in light of the rule of law.

David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. 2019. Xai—explainable artificial intelligence. *Science robotics*, 4(37):eaay7120.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. 2020. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040.

Erik Knudsen. 2023. Modeling news recommender systems' conditional effects on selective exposure: evidence from two online experiments. *Journal of Communication*, 73(2):138–149.

Max Kuhn and Kjell Johnson. 2013. *Data Preprocessing*, pages 27–59. Springer New York, New York, NY.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized transformer for explainable recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4947–4957, Online. Association for Computational Linguistics.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Leland McInnes, John Healy, and James Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mitchell Naylor, Christi French, Samantha Terker, and Uday Kamath. 2021. Quantifying explainability in nlp and analyzing algorithms for performance-explainability tradeoff. *arXiv preprint arXiv:2107.05693*.

Britta Oertel, Diego Dametto, Jakob Kluge, and Jan Todt. 2023. *Algorithmen in digitalen Medien und ihr Einfluss auf die Meinungsbildung*.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Ammar Rashed, Mucahid Kutlu, Kareem Darwish, Tamer Elsayed, and Cansın Bayrak. 2021. Embeddings-based clustering for target specific stances: The case of a polarized turkey. In *Proceedings of the International AAAI Conference on web and social media*, volume 15, pages 537–548.

Shaina Raza and Chen Ding. 2022. News recommender system: a review of recent progress, challenges, and opportunities. *Artificial Intelligence Review*, pages 1–52.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Amy Ross Arguedas, C Robertson, Richard Fletcher, and R Nielsen. 2022. Echo chambers, filter bubbles, and polarisation: A literature review.

Gabriella Vas. 2022. Personalization for news sites: How news personalization keeps readers engaged.

Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Npa: Neural news recommendation with personalized attention. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2576–2584.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online. Association for Computational Linguistics.

A Appendix

Constructing interpretation from values: We visualize the retrieved values using wordclouds. The *wordcloud module* filters words by only considering the values of positive articles, meaning articles that have a recommendation score of 0.5 or higher. From each headline, we retain only the three words with the highest value. Finally, we then filter so-called stopwords such as “and” and “in”, as well as words shorter than three characters. Given the remaining words and their respective values, we test two approaches:

1. **Scaling:** We scale the value of each word by the recommendation score of its headline, such that words of weakly recommended articles (score of 0.5) have less weight than words of strongly recommended articles (scores close to 1.0). We then add up all word values.

2. **Counting:** We count the occurrences of each word regardless of its value and use this word frequency for display.

Of the two approaches, the *scaling*-approach yields more balanced and representative wordclouds. A comparison of the two different wordclouds for one user embedding can be found in Figure 7.

(a) Attention counting (b) Attention scaling (c) LRP Counting (d) LRP Scaling

Figure 7: The wordclouds for the same user embedding as generated by using the attention values and the LRP relevancy values. “Counting” means that word occurrences were counted regardless of the strength of the recommendation, while “Scaling” scales the words according to the recommendation value of it’s headline