

Universiteit Leiden

Master Computer Science

Personalized and efficient neural news recommendation using semantic embeddings as model input

Name: Jelle Keulemans
Student ID: 2357682

Date: 10/07/2026

Specialisation: Data Science

1st supervisor: Suzan Verberne
2nd supervisor: Marlies van der Wees

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Two-tower neural news recommenders (NNR) heavily utilize article content features for recommendation because of the short article relevance window. For this reason, NNRs utilize a news encoder to encode article content, and a user aggregator that combines user reading history. How the news encoders in NNR are trained, largely defines the efficiency and accuracy trade-off, because their time complexity scales with input length and because they commonly include hundreds of millions of pre-trained model parameters to achieve good performance. We propose Semantically Adapting Neural News Recommender (SANNR) that explores a novel trade-off: we design the NNR architecture to perform task adaptation on semantic embeddings generated by pre-trained language models (kept frozen), allowing for permanent caching of the embeddings, and efficient training, while achieving improved recommender performance compared to similarly efficient NNRs. To achieve this, we designed a combined architecture from NNR, regularization, and sequential recommender components, which we all ablated to identify well-suited variants. SANNR outperforms NAML, NRMS, LSTUR, and MINER with frozen news encoders on MIND and our dataset: **NU**.nl **ME**dia **N**ews **D**ataset (NUMEND). Furthermore, it achieves 10% increased nDCG@10 ranking performance and 30% increased precision@10 classification performance compared to a non task-adapted news encoder, while achieving efficient training (up to 20,000 samples per second), and inference (up to 50,000 samples per second), which is 100× faster than end-to-end trained NNRs. We found that SANNR can be optimized for recommendation by: (i) modeling user behavior, (ii) detrending popularity, (iii) remaining time-invariant, (iv) optimizing for all user activity groups, and (v) optimizing for news categories that most benefit from personalized news recommendation. An inter-user study with 12 participants showed significantly ($p < 0.05$) increased precision but reduced diversity compared to non task-adapted semantic recommendation.

1 Introduction

Personalized news recommendation suggests news articles to users they might want to read based on user features such as reading history. This helps users sift through vast news catalogues of hundreds of articles published per day. Unlike most other recommendation domains such as e-commerce, most news articles are only relevant for a short period of time (e.g. one to two days on the Dutch nu.nl news platform), meaning that they need to be recommended soon after publication with no chance to have accumulated many user interactions, which is known as the *item cold-start problem* (Meng et al. 2023; Wu et al. 2023). This motivates modality-based recommendation in the news domain where article representations are derived from the textual article content rather than user interactions that are sparse for newly published articles (Yuan et al. 2023). Our work focuses on two-tower neural news recommenders (NNRs) such as NRMS (Wu et al. 2019c) and MINER (Li et al. 2022) that encode users and

items separately, allowing for caching, and thus efficient ranking of the full item catalogue at the speed of dot products.

NNRs all need to encode textual news article data into numerical representations, and this design choice determines the ball-park of recommendation accuracy, training compute, and inference compute (Iana et al. 2025; Li et al. 2025). Inference compute here is mainly determined by item cache preservation and cache filling compute. **On one side of the accuracy and compute spectrum**, article embedding representations are created by pre-trained sentence encoders and permanently cached, and candidate article relevance to user histories is measured as the average similarity of the candidate representation to user history embeddings (Li et al. 2025). This approach is parameter-free and is commonly referred to as: 'late fusion', because user representations can be efficiently created on-the-fly. Late fusion is the simplest and cheapest approach because it is just an averaging operation, and generalizes to any unseen article, regardless of publication timestamp. A limitation of this approach is that it performs no task adaptation on click data, leaving it unoptimized for: (i) shifting trends, (ii) news platform-, language- and region-specific news texts, (iii) news popularity, (iv) news categories, and (v) (platform-specific) reading patterns[1], reducing recommender accuracy (Li et al. 2025; Huang et al. 2023). Furthermore, without parameter optimization, it is for instance more difficult to create ensembles, impossible to add arbitrary loss terms for multiple objectives, and hard to make use of more input features. **On the other side of the spectrum**, sentence encoders are fine-tuned jointly with NNR using click data. This is the most accurate option within two-tower NNR literature (Wu et al. 2021), but also the most expensive to train, and the model must be frequently re-trained to account for shifting news trends (Xiao et al. 2022), and invalidates all news embedding caches after each training cycle. A cheaper but less accurate variant trains a shallow news encoder from scratch end-to-end, which is the original methodology for NNR such as NRMS (Wu et al. 2019c) that don't (yet) leverage BERT encoders.

We explore a sweet spot on this spectrum: keeping the sentence encoder frozen and adding a trainable set of NNR components that optimizes item embeddings and user representation for recommendation. This approach is the cheapest way to achieve task adaptation as it skips out on optimization of item encoder parameters and only optimizes trainable components on top of the now fixed embeddings. So, this also means that item embedding caches are never invalidated. Iana et al. (2025) explored this approach but report a negative result. They show that the trained NAML NNR with a frozen news encoder does not outperform a parameter-free late fusion recommender with a sentence encoder that is fine-tuned on in-domain news texts. However, we argue that this nega-

[1]On nu.nl we see that articles about actor gossip and movies have semantically similar embeddings (because of the shared actor name) while users do not frequently interact with both (diverse interests). We also see the opposite effect where articles of movies and books for example are encoded as semantically diverse embeddings while users frequently interact with both.

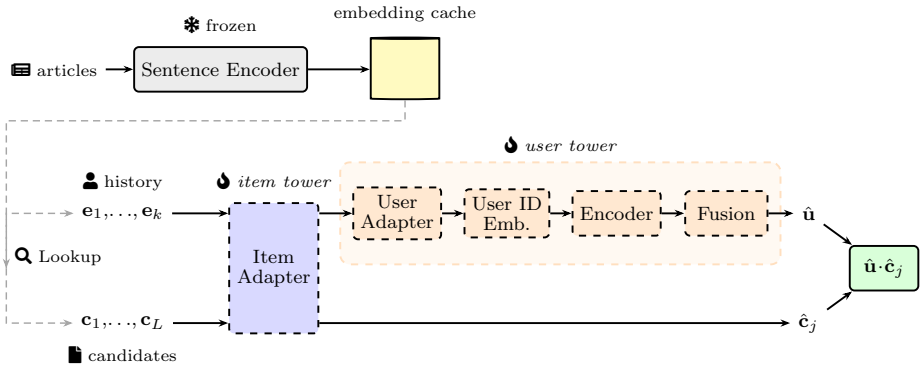


Figure 1: Architecture of SANNR. Article embeddings are produced once by a frozen sentence encoder and stored in an embedding cache; during training and inference, history and candidate embeddings are retrieved from the cache (dashed lookup paths) rather than recomputed. Dashed components are individually toggleable for ablation; the item adapter shares weights across both towers. When all dashed components are disabled, SANNR reduces to the late fusion baseline.

tive result for frozen NNR only implies that NNR architectures such as NAML are not designed for optimizing over frozen item encoders; they are designed to be trained jointly with token-level news encoders. Freezing the news encoder results in fewer input features: only one embedding per article, meaning that the bias-variance balance shifts, and that regularization measures and different model architectures may be more suitable for task adaptation without token-level regularization. Iana et al. (2025) do not consider this. Moreover, to the best of our knowledge, no papers have considered this limitation with frozen NNR.

NNRs designed for training on top of frozen news encoders are underexplored. Specifically, no efforts are reported to add trainable components that learn semantic embedding projections to achieve task optimization, even though this is common practice in sequential recommenders such as DropoutNet (Volkovs et al. 2017), GAR (Chen et al. 2022), AlphaFuse (Hu et al. 2025), the delta embedding approach of Pembek et al. (2025) and ALDI (Huang et al. 2023) that all learn a mapping from frozen semantic embeddings to embeddings task-optimized with behavioral signal. It is unknown if such projection layers can be effective for frozen NNRs, which NNR user modeling components are effective, and if regularization helps in this domain. To this end, we propose SANNR (Semantically Aligning Neural News Recommender), a modular recommender that implements embedding adaptation (Pembek et al. 2025), self-attention history encoding (Wu et al. 2019c), poly attention history fusion (Li et al. 2022), learned user-ID embeddings (An et al. 2019), and regularization techniques (Lu 2020; Jain et al. 2023) that can all be ablated independently (Figure 1). Although

this underperforms end-to-end NNRs, it preserves embedding caches (which is a common requirement in production environments) and requires a fraction of the compute for training and cache filling, meaning that it is as efficient as the late fusion recommender. Furthermore, we identify SANNR configurations that outperform the late fusion recommender and all tested NNRs applied to static embeddings on the MIND dataset, and on a newly sampled dataset from the nu.nl news platform (NU.nl MEdia News Dataset: NUMEND) that contains more region- and locale-specific Dutch news.

Research questions We address four research questions:

RQ1 Semantic baseline: How do article encoding strategies (such as which text to use and how they are pooled) and the sentence encoder family affect zero-shot personalized news recommendation using the semantic embeddings without task optimization steps?

RQ2 Architecture for frozen news encoders: Which lightweight news and user modeling components (and component combinations), drawn from NNR and sequential recommendation literature, add the biggest improvements to recommendation from static semantic user history embeddings as model input? And to what extent do modifications for handling frozen news encoders (such as embedding projection) improve over (i) the parameter-free semantic baseline and (ii) established NNR architectures (e.g. NRMS, LSTUR, MINER, NAML) applied to frozen embeddings?

RQ3 Task adaptation characteristics: Considering user behavior, popularity bias, news categories, user history length, and generalization to future dates on nu.nl; in what ways does SANNR's task adaptation change recommendation behavior compared to the frozen semantic baseline?

RQ4 User study: Do users perceive personalized recommendations by SANNR as more relevant and interpretable compared to the semantic recommender?

Contributions Addressing the research questions, our contributions to the literature are as follows:

1. We construct NUMEND, a MIND-style Dutch news recommendation dataset sampled from nu.nl, enabling evaluation on a different locale than MIND, also allowing us to answer the research questions in a practical context;
2. Addressing RQ 1, we perform a large-scale evaluation of common article encoding strategies and open-source sentence encoders on the recommendation task to find the most effective news encoding configurations for late fusion recommendation;
3. We propose a novel and modular neural news modeling architecture for addressing RQ 2, 3 and 4: Semantically Aligning Neural News Recommender

(SANNR), which includes ablatable trainable components inspired by related work for task adaptation, each designed for optimizing over *static* news embeddings;

4. Addressing RQ 2, we ablate all SANNR components to find the best components for personalized news recommendation ranking and classification performance with frozen news encoders, and to quantify each component's individual contribution;
5. Addressing RQ 2, we compare SANNR's recommendation performance and training and inference time against the user-modeling components of NAML, NRMS, LSTUR, and MINER applied on top of frozen encoders, and against reported results for shallow end-to-end-trained NNR;
6. Addressing RQ 3, we conduct detailed analyses on NUMEND measuring popularity bias, temporal overfitting (on trends), performance across categories and user history lengths, and how SANNR's task adaptation models user behavior, including how its predicted similarity aligns with co-click behavior;
7. Addressing RQ 4, we evaluate the most efficient variant of SANNR in a user study to further prove that offline performance increments translate to perceivable improvement in personalized news recommendation.

2 Related work

This section first establishes collaborative filtering methodology, which relates to the trainable user-ID embeddings in SANNR; then embedding task-adaptation methodology from the sequential recommendation domain; next personalized news recommendation, including work that focuses on efficiency close to our work. We finish with positioning our scientific contributions within the related work.

2.1 Collaborative filtering

In collaborative filtering (CF), user ratings or clicks are predicted by users that interact with similar items, or by items that are interacted with by similar users (Schafer et al. 2007). Latent factor models extend on this by decomposing binary or categorical interaction data to latent representations, where user and item IDs map to embedding tables that factor together to predict the full user-item interaction matrix. Common approaches to optimize these embeddings are Alternating Least Squares (ALS, Takács et al. (2012)), Singular Value Decomposition (SVD, Zhang et al. (2005)), and Funk Singular Value Decomposition (Funk-SVD) that uses stochastic gradient descent to optimize the embeddings (Paterek 2007). SANNR follows a similar approach to Funk-SVD, but optimizes using a different loss function, and does not optimize user-ID

embeddings jointly with item-ID embeddings. Model-based collaborative filtering methods such as Neural Collaborative Filtering (He et al. 2017) extend on latent factor models by adding trainable components on top of collaborative embeddings. This is more similar to the user-ID embeddings implementation in SANNR than Funk-SVD, because SANNR implements self-attention and additive attention over the embeddings. Doing so, content-features can be effectively combined with collaborative embeddings (Papadakis et al. 2022). However, collaborative filtering models cannot effectively handle cold-start items, changing user preferences, and can be inefficient to train because they optimize over the full user-item interaction matrix. SANNR is a hybrid between content-based methods and collaborative-based methods, but heavily leans towards content-based to tackle the challenges with collaborative filtering, while modeling some user-specific collaborative signal.

2.2 Neural news recommendation

NNRs are designed for cold-start recommendation and generalization to future interest distributions, relying mainly on content features. Two-tower news recommenders are the most efficient type of NNR. In two-tower news recommendation, news modeling and user modeling are split, and not dependent on candidate articles to allow for fast relevance scoring using for example dot products. News modeling encodes content of articles in such a way that similar embeddings correspond to articles that are liked by the same type of users, and user modeling encodes complex interests of users derived from their user histories such that users with similar interests get similar embeddings. This is similar to how CF learns ID embeddings, but in a much more restricted way, also requiring a different training and inference task type.

News modeling Early NNR such as LSTUR (An et al. 2019), NPA (Wu et al. 2019b) and NAML (Wu et al. 2019a) implement news modeling with CNNs to extract local context from tokens, which they combine into one embedding using additive attention. LSTUR and NAML also concatenate category embeddings to the news embeddings and show this increases nDCG@10 ranking performance on MIND by about 1%. NPA proposes the personalized additive attention where a learned user ID-based preference query attends over the tokens, which again slightly boosts ranking performance by about 1%. NRMS (Wu et al. 2019c) replaces CNN with multi-head self-attention (MHA) to capture global token relationships instead of local n-grams that convolutional layers capture, again slightly improving recommender performance. Wu et al. (2021) showed that all of the above 'shallow' news encoders could simply be replaced with a pre-trained language model (PLM) such as RoBERTa to improve their performance by a large margin (about 5% increased nDCG@10 ranking performance on MIND) when fine-tuned with NNR user modeling and loss. MINER, created by the same research group, uses classification token output from a pre-trained BERT model, outperforming all other news encoders (Li et al. 2022).

User modeling LSTUR implements user modeling with a Gated Recurrent Unit (GRU) layer that is initialized with a user ID embedding capturing long-term interest (and user-collaborative signal), which they also apply dropout on, similar to DropoutNet. They show that ID embeddings are valuable in modeling long-term user preferences and increase recommender performance. Because GRUs are hard to parallelize, NRMS replaces this with news-level MHA and additive attention, treating click history as unordered. Furthermore, they drop user ID embedding tables because they require more memory. Even though NRMS does not use ID-based user embeddings and is cheaper at inference, it achieves comparable performance to LSTUR by considering global entity relationships instead of local relationships through GRU or CNN layers. NAML does not model inter-item relationships and only uses additive attention to create user embeddings, which performs slightly worse than NRMS. NPA and HieRec take the different route of personalized additive attention over history embeddings, outperforming NAML and NRMS. MINER improves on additive attention with poly attention where multiple types of user interests are extracted by attending over user histories with K global codes to create K user embeddings. MINER outperforms all the above NNR on the MIND dataset. Further candidate-aware patterns where candidates are included into the user embeddings invalidate user embedding caches for every new candidate, so we do not include them.

2.3 Efficient two-tower neural news recommenders

As mentioned in the introduction, literature shows that there is a trade-off when incorporating PLM in NNR. On the one hand, they could be kept frozen, but then they underperform shallow NNR because they implement no task adaptation. This is the approach SANNR explores. On the other hand, they could be trained end-to-end with NNR to achieve much better accuracy at the cost of expensive training and cache refilling. Allowing for efficient NNR that harness the strength of PLM, one area of research focuses on improving task adaptation of frozen sentence encoders, while the other focuses on the engineering of efficient pipelines that allow for end-to-end training of sentence encoders with NNR.

Efficient end-to-end training Two lines of work stand out that explore efficiency of end-to-end trained NNR. Xiao et al. (2022) propose engineering tricks such as intelligent caching and distributed computing to be able to do full PLM fine-tuning. Iana et al. (2023) propose to implement user encoders for late fusion with the goal of slightly reducing required training compute and architectural engineering complexity so that it becomes easier to keep PLM in the loop.

Task adaptation of frozen encoders Iana et al. (2025) fine-tune a sentence encoder for the news domain (NaSE) to achieve some level of task adaptation.

They show that this sentence encoder with late fusion user modeling achieves better performance than this encoder with NNR user modeling on top. Furthermore, evaluated on MIND, NaSE with late fusion matches the performance of an unoptimized sentence encoder whose embeddings are task-adapted by NAML user modeling components. So, evidently, NAML achieves some level of task adaptation that is equivalent to NaSE, in a very lightweight way, which the authors don't directly acknowledge. Moreover, they don't optimize NAML for handling embeddings as model input. To the best of our knowledge, there is no further related work in this domain, highlighting the gap in the literature that we position against: optimizing NNR for task adaptation of embeddings generated by unoptimized frozen sentence encoders.

2.4 Sequential item cold-start recommenders

Sequential recommenders predict the most likely next user interactions given a list of previous interactions. Unlike NNR literature where pre-trained item encoders are commonly fine-tuned to adapt to behavior space to allow for cold-start item recommendation, content-based sequential recommenders usually align content encodings with behavior space in other ways, without full fine-tuning of content encoders. DropoutNET combines ID embeddings with modality embeddings by applying dropout on the ID embeddings during training, forcing the model to predict behavioral patterns from content (Volkovs et al. 2017). GAR (Chen et al. 2022) trains an adversarial network to generate synthetic ID embeddings from content embeddings. ALDI (Huang et al. 2023) aligns content embeddings for recommendation by using an ID-based recommender as a teacher model for the content-based recommender to map the parameters to behavior space. TIGER (Rajput et al. 2023) takes a different approach of quantizing embeddings to semantic tokens using a variational auto-encoder to allow LLMs to map them to a behavior space leveraging their knowledge. Additionally, some efforts are made to add collaborative signal in semantic embeddings through a constrained learnable delta that is added to semantic embeddings (Pembek et al. 2025), and adaptation of learning tasks in 'zero-space' of semantic embeddings that carry little semantic signal (Hu et al. 2025). Hu et al. (2025), Volkovs et al. (2017), and Chen et al. (2022) show that it is possible to train content embedding projections for task adaptation. Hu et al. (2025) even show that freely optimized ID embedding tables can be replaced by projection layers. In this work, we venture into adding a collaborative signal to content embeddings in a constrained way, similar to Hu et al. (2025) and Pembek et al. (2025), that add learnable projections to semantic embeddings without 'destroying' semantic space.

2.5 Contributions

In SANNR, we combine user modeling components of NRMS, MINER, LSTUR and NAML, and trainable delta embedding adaptation into one modular architecture. Doing so, this architecture contains a diverse set of components that

can each achieve task adaptation in a distinct way, allowing us to uncover an effective architecture for the frozen NNR domain in an ablation study. Specifically, the components that we combine are: self-attention from NRMS (leverages history sequence dependencies), poly attention from MINER (extracts multiple user embeddings), user ID embeddings from LSTUR (learns true personalized embeddings per user), personalized attention from NAML (learns personalized history aggregation per user), and delta embedding projection (optimizes semantic embeddings for recommendation without touching item encoder weights). Furthermore, we adapt some components for the frozen NNR domain (for instance testing projection-free self-attention), and add regularization measures to explore whether they are effective in this frozen NNR domain.

We position SANNR against news-fine-tuned sentence encoders such as NaSE, trained unadapted NNR with frozen news encoders, and shallow end-to-end trained NNR. Furthermore, we find that in the literature, recommenders are often only evaluated on the MIND dataset (sampled from Microsoft News) using the nDCG@10 ranking metric. However, Microsoft News is a very different platform from many other news platforms such as nu.nl [2], for which the news texts are more out-of-distribution for any sentence encoder. Covering these limitations in model evaluation, we test SANNR on an additional dataset sampled from nu.nl click logs. Furthermore, we do a detailed analysis to uncover how SANNR performs task adaptation, including an analysis that specifically explores user behavior modeling, and a user study that explores users' experience with the recommender systems.

3 Data

In this project we construct a dataset from impression logs on nu.nl from DPG media, which we call Nu.nl MEdia News Dataset (NUMEND). This dataset allows for offline evaluation and training of personalized news recommender systems in Dutch on one of the most high-traffic Dutch news platforms. The nu.nl news platform has unique characteristics compared to platforms like Microsoft News such as a human-curated home page and a small, high-quality, moderated, same-source set of articles, making it a suitable dataset to evaluate benchmark recommender performance on this more niche task and practical use-case.

In this section, we describe how we sample NUMEND from log data, and compare the dataset to the MIND (Wu et al. 2020) benchmark dataset across various metrics. Afterwards, we do an exploratory analysis to identify requirements for user modeling for personalized news recommender systems in the NUMEND dataset, and limitations of embedding similarity methods for recommendation. The exploratory analysis specifically covers

1. Article metadata distributions and general data statistics (NUMEND com-

[2]We for example find that the datasets differ in content language, writing style, article count, interaction sparsity, exposure bias, article relevance window, and user behavior.

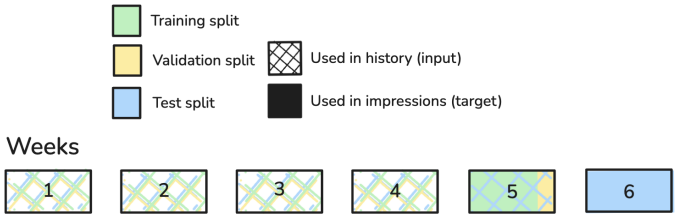


Figure 2: Illustration of the 6-week train/validation/test data split. The training set uses Week 5 impressions with history from Weeks 1-4. From this set, the validation is split using the impressions from the last day of Week 5. The test set uses Week 6 impressions with history from Weeks 1-5.

pared to MIND)

2. User cold starts (NUMEND compared to MIND)
3. Popularity bias log analysis (currently only on MIND)
4. User interest patterns such as interest change and trend-following (only on NUMEND)
5. Co-occurrence and conditional probability of articles versus their semantic similarity (only on NUMEND)

Afterwards, we propose the direction for appropriate methodology for news recommendation on NUMEND, with a focus on user modeling, which addresses the patterns that we identified in the exploratory analysis.

3.1 Preliminary: MIND sampling

MIND is constructed from impression logs containing the IDs of articles that users see, and whether or not they were clicked (the labels). The format is: "`article1-1 article2-0`", where articles with IDs `article1` and `article2` are sequentially observed by a user, and `article1` is clicked. Each impression list is paired with a timestamp, user history (articles that users clicked before) and a user ID. Metadata for each article consists of the abstract, title, author-defined category, author-defined subcategory, and body (which is missing because the links expired). So the news recommendation task in this dataset is to click labels of impression logs given the user history, article metadata and user ID.

The logs are sampled over a 6 week period. From these logs, 1 million users with 5 or more news clicks are randomly sampled, and anonymized. The MIND-small version instead uses a subsample of 50 thousand users.

MIND is split using a sliding-window approach as shown in Figure 2. Each split consists of article impression logs for a defined period, and user history of user

article interactions that happened *before* this period. So, in each data split, this history sequence is constant per user.

As illustrated in Figure 2, the training split samples the user click history from the first four weeks, and includes impression logs of the last week. A subsample of the training set of the last day creates the validation set. And the test set uses user history of weeks 1-5 and the impression logs of week 6. Test split labels are private to accommodate fair centralized standardized test performance evaluation in the MIND leaderboard.

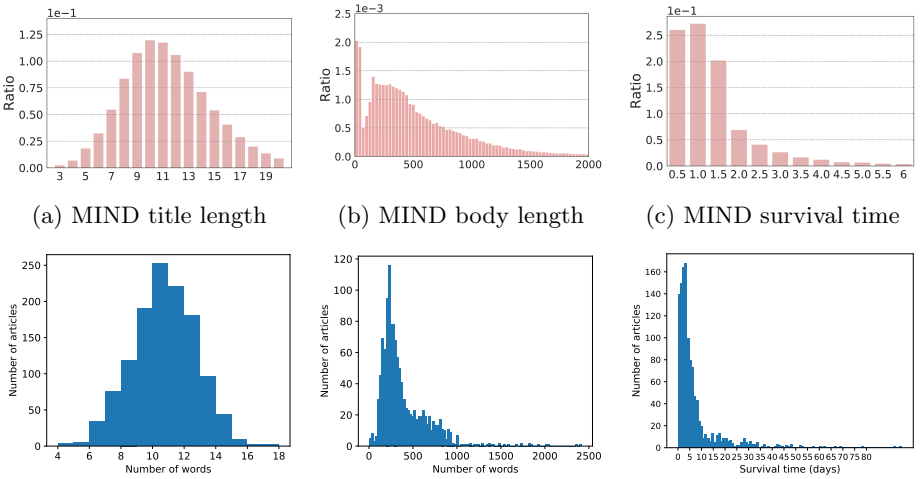
For our project, we use the MIND-small split because this requires less memory to load and less compute to evaluate. Because testing labels are not available, we split the training split into a training split and validation split by taking the last 2 days for validation, and use the dev split as testing split. So, all references to MIND refer to the small subsample unless stated otherwise.

3.2 Sampling NU.nl Media News Dataset (NUMEND)

NUMEND is sampled in a similar fashion as MIND, but from click logs instead of impression logs, sampled over a shorter time period and using a smaller subsample of users to save compute. Only users with 3 or more clicks during the three week sampling period are included. For these users, 10% of users are randomly sampled. Also, we only keep clicks with more than 30 seconds of 'dwell time', so that we are sure that users have actually read the article, and we aggregate consecutive clicks into one session that creates an impression log of a number of clicked articles.

The training split includes user history from week 40 of the year 2025, and *click log* targets from the first 5 days of week 41. Impression logs are unavailable at DPG Media, so we use negative sampling to sample negative impressions. Doing so, the log format looks identical to MIND's impression log format. The validation split uses the same history as the training split, and uses the *impressions* from the last 2 days of week 41. The testing split uses the history from weeks 40 and 41, and labels from week 42.

We sample negatives from the popularity distribution, meaning that the most popular items are sampled as negatives most. We do not sample with replacement. Sampling articles with the probability derived from article popularity reduces the chance that users would have clicked the negative sample if they had seen it because the chance is higher that users see popular articles, especially on nu.nl. Therefore, we think this is a valid replacement for impression logs, where unseen impressions also follow the popularity distribution. We compute popularity by dividing the number of clicks on each article in the last 6 hours by the total number of article clicks in the last 6 hours. The time difference uses the time stamp of the positive click we sample negatives for as a basis. We think that this negative sampling strategy is essential on nu.nl since this punishes recommendations using just article popularity, and forces any recommender to infer personalized user interests.



(a) MIND title length (b) MIND body length (c) MIND survival time

(d) NUMEND title length (e) NUMEND body length (f) NUMEND survival time

Figure 3: Dataset distributions for MIND (top row, copied from Wu et al. 2020) and NUMEND (bottom row): title length, body length, and article survival time. Survival time is the number of days after publication that an article is read at least once.

Article metadata almost has full overlap with MIND's article metadata. The only difference is that NUMEND includes no article abstract and that MIND includes no article body (because the references to them expired). We consider both as 'article content'.

3.3 Dataset summaries

Subfigures 3a, 3b, and 3c respectively show the title length, body length and survival time distributions of MIND, reported by Wu et al. 2020. The bottom three subfigures in Figure 3 show the three distributions in NUMEND. The survival time is given by the amount of days that an article is read at least once after publication, also known as the article relevance window.

The title length distributions in figures 3a and 3d differ in that NUMEND has a much more narrow distribution where the deviation between title lengths is smaller. Article titles by nu.nl are likely more consistent than article titles on Microsoft news because they are from the same source and all managed professionally by the same team.

Figures 3b and 3e show the body length distributions in MIND and NUMEND respectively. From this we observe that MIND's body length distribution is split between one portion of the articles with very short body length of shorter than 40 words, and one portion with much longer lengths of over 200. We don't see a split body length distribution in NUMEND. Ignoring this portion of smaller

Table 1: Summary of a number of statistics for the MIND, MIND (small) and NUMEND datasets. Values for MIND (full) are copied from Wu et al. 2020. MIND (small) is a subsample of 50 thousand users of MIND. Training and validation splits are sampled from the MIND (small) training split, and testing split is the validation split.

	MIND (full)	MIND (small)			NUMEND		
	All	Train	Val	Test	Train	Val	Test
Users	1,000,000	46,012	20,179	50,000	25,458	19,285	30,620
Articles	161,013	16,978	6,144	5,369	579	548	1,123
Impressions	15,777,377	126,695	30,270	73,152	95,851	60,130	186,459
Categories	20	18	-	-	5	-	-
Subcategories	-	270	-	-	35	-	-
Avg. title len.	11.52	10.74	-	-	10.24	-	-
Avg. abstract len.	43.00	36.79	-	-	-	-	-
Avg. body len.	585.05	-	-	-	318.16	-	-
85%ile survival (days)	2	2	-	-	15.22	-	-

length articles in MIND, we see that in both datasets, a body length of around 250 words is most common, but that on average, articles in MIND are longer than in NUMEND. Lastly, we see that in MIND, the distribution stretches out much more towards higher body lengths than in NUMEND. In NUMEND it is very uncommon to see articles of 1000 or more words, while in MIND this is much more common. This is again likely the result of more consistency in news platform content.

Figures 3c and 3f showcase the survival time of articles in both datasets. Here we see the largest difference between the two datasets. Articles in MIND 'survive' for only 4 days for almost all articles, while articles in NUMEND may 'survive' much longer, even up to 10 days for most articles. And the most frequent survival time is 1 day in MIND, and 5 days in NUMEND. The long tail of survival times in NUMEND is also much longer, and even stretches to beyond this figure. So nu.nl includes articles that people reference back to after weeks, months or even years. So in NUMEND there is a slight reduction in item cold-start because articles stay relevant longer, with the majority of articles staying relevant 5 times longer (10 vs 2 days). However, item cold-start is still clearly one of the main challenges in NUMEND.

Table 1 summarizes statistics of both datasets. In summary, notable differences are that MIND has more users (even the small split), articles and articles in every split. MIND has more impressions per user in the training split, but less in the validation and test splits. Furthermore, MIND has several orders of magnitude fewer impressions per article in all splits. This result is expected since NUMEND is sampled from a news platform with fewer article publications per day. One further difference stands out: the survival time is much longer for

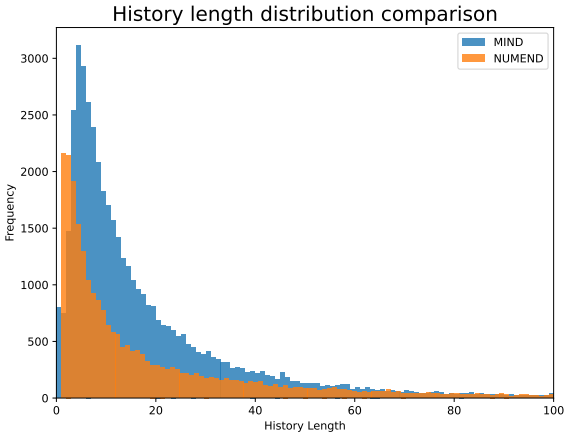


Figure 4: Distribution of counts of users over their history length in the training splits of the MIND and NUMEND datasets, showing a right-skewed distribution where most users have short history lengths of lower than 10 while fewer users have a longer history length.

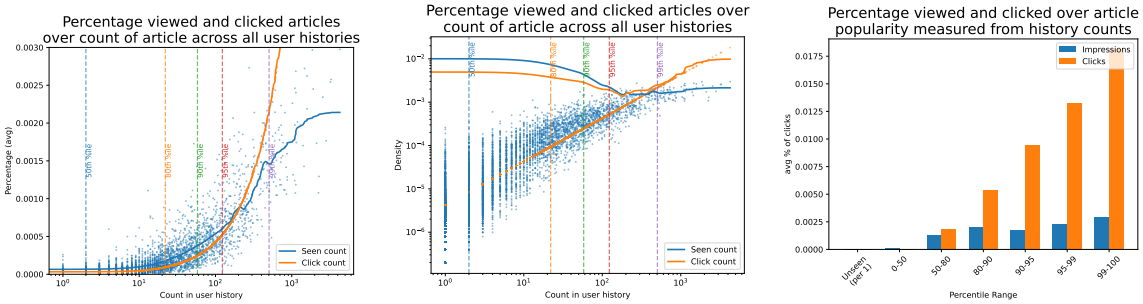
articles in NUMEND. 85% of published articles are relevant for at least some users for two weeks for NUMEND, while only 2 days for MIND. It is clear that some users revisit articles on nu.nl. We hypothesize that this is caused by sharing behavior causing users to revisit articles; we don't know why people aren't revisiting articles on Microsoft News, and suspect undocumented filtering by Wu et al. (2020).

3.4 History length distributions

Figure 4 illustrates the history length distribution over users for the training splits of MIND and NUMEND. Table 2 shows the exact history length over activity percentiles, where the percentile corresponds to the percentage of users that have an equal or a shorter history length. From this figure it's clear that both distributions are right-skewed: most users have very short history lengths while only a few users have longer lengths. It is also heavy-tailed: user counts drop steeply for smaller history lengths and drop gradually for large history lengths. This confirms the user cold-start challenge in (news) recommendation where most users are inactive and few are very active (power users), causing epistemic noise in predicting preferences for inactive users and overfitting on preferences of power-users. From the table and figure, it's clear that users in MIND have longer histories at the lower percentiles (up to the median), while NUMEND is heavier-tailed with longer histories from the 75th percentile onward. Notably, the difference in frequency becomes larger as the history length decreases, showing that MIND is more skewed toward short histories.

Table 2: User history lengths over the percentage of users that have smaller history lengths.

	5%tile	25%tile	50%tile	75%tile	80%tile	90%tile	95%tile	99%tile
MIND history length	2	5	11	23	28	44	63	119
NUMEND history length	1	4	10	27	34	58	85	149



(a) Averages (b) Cumulative (c) Percentile Distribution

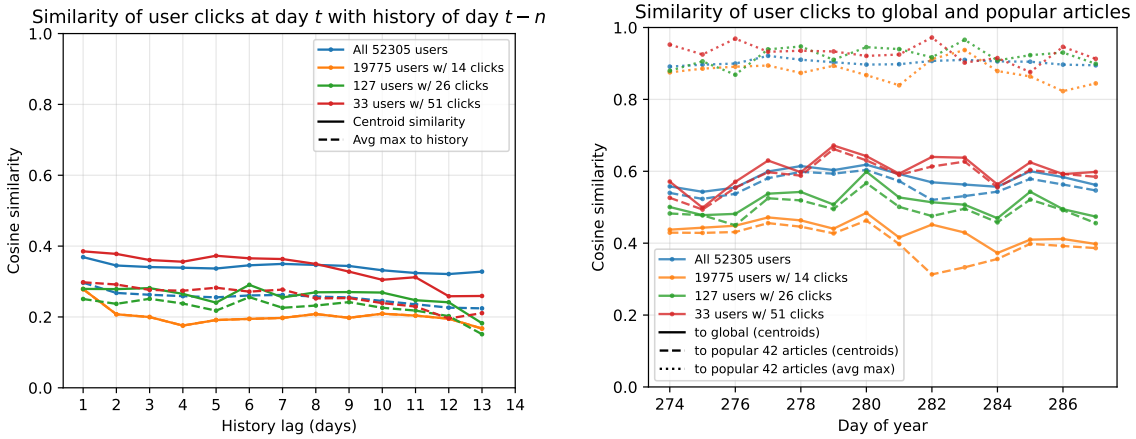
Figure 5: Article popularity bias analysis in the MIND Dataset. Popularity is measured by the count in histories. (a) Shows the relationship between article impression percentages and click percentages across article popularity. (b) Shows the cumulative percentages. (c) Shows averaged impression and click distributions by popularity percentiles. It shows that there is a bias towards recommending too many non-popular items (0-50 percentiles) and too few very popular items (larger than 50th percentile).

In the figure, we clearly see an artifact on the left hand side caused by filtering: lower history lengths are cut off, and the frequency drops near the history length of 0 as it becomes less likely for users to have all their clicks concentrated in the testing split, which is not included in the figure and table.

3.5 Popularity bias

The distribution of popularity of news articles in the literature usually is long-tail where few articles are very popular and most are not popular. A popularity bias in recommender systems occurs when popular items are disproportionally recommended over non-popular items to the degree that the quality of the recommendations decreases (Abdollahpouri et al. 2021). Recommendations can become (i) not diverse, (ii) not personalized, (iii) irrelevant, (iv) unfair. The goal of this analysis is to identify bias in articles that were served to users in the impression logs. This means that we currently only conducted this analysis on the MIND dataset because we have no real impression logs for NUMEND for identifying popularity bias on the existing system.

We define article popularity as percentiles, where the percentile refers to the



(a) Interest Change

(b) Trend Following

Figure 6: Similarities between interest distributions, including similarity between current interest and interest several days ago showing interest drift and varying trend-following behavior over days. Several subgroups are included: 20th percentile active users, 50th percentile active users and the most active 80%tile active users that are more active than 20, 50 and 80 percent of users, respectively. Each subgroup is a slice of users with an equal amount of clicks. The most active users have the highest similarity scores and the least active the lowest.

combined percentage of clicks that the bin receives. So the N articles in the 80th percentile together receive 80% of user clicks.

Figure 5 shows article exposure to users and the user preferences over the popularity of articles measured in counts in user histories. In Figure 5a, the percentages of the total amount of articles seen and clicked are shown over the count of articles in combined user histories. Per history count, the percentages are averaged. When comparing the click count and impression counts over article popularity, it is clear that users interact with popular items more (Clicks) than the in-place recommender served them (Impressions). This indicates that there is no *global* popularity bias in the impression logs where people are served more popular items than they like to interact with. However, this does not rule out local per-category popularity bias in the impressions.

3.6 User interest analysis

In this section, we analyze how user interests change through time, the overlap between user click behavior and popular articles and which fraction of article reads is explained by semantic similarity to user history. Only results for NUMEND are included because MIND only has one week of labeled log data.

The goal of this analysis is to identify (i) shifting user interests, (ii) trend-following interests and (iii) ratios of history-driven reading over user counts and categories. Learning about these properties can help to identify requirements for NNR architectures such as the 'relevance window' of user interests and the effectiveness of content-driven recommendations across categories and users.

Learning about interests is complex: it is the reason that much literature shifts to using deep learning architectures such as transformers that can capture interest co-occurrence patterns (Meng et al. 2023). The drawback of using deep learning methods for modeling user interests is that it is hard to interpret: they are black box algorithms. So, for this data analysis, we follow a simpler approach where we compute semantic similarity scores between sentence transformer generated embeddings, inspired by Liu et al. (2010). They compute differences between category-based interest vectors. However, instead of using category-based vectors, we use the MPNet-multi sentence transformer from Song et al. (2020) to generate semantic embeddings from article title and body because this creates representations that more closely match content encoding of NNR that we explore, allowing us to identify interest patterns that NNR can capture using semantic embeddings.

Given a set of embeddings for articles clicked by user u at time t $C_t(u) = \{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$, and the set of embeddings for the target article collection $D_t = \{\mathbf{o}_1, \ldots, \mathbf{o}_m\}$, we compute two metrics to compare these sets: (i) the **Centroid Cosine Similarity**, defined as $\cos(\mu_u, \mu_d)$, where μ_u and μ_d are the mean vectors of $C_t(u)$ and D_t, respectively; and (ii) the **Average Maximum Similarity**, defined as $1 - \frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{o}\in D_t} d(\mathbf{e}_i, \mathbf{o})$, which measures the average distance from each article in the user's history to its nearest neighbor in the target set using a distance function $d(\cdot)$, for which we use cosine similarity. When computing these metrics at time t across users, we first compute the metric for each individual user, and then average the metrics across users, ensuring that active users don't dominate the similarity scores.

Figure 6 shows similarities between user interests at time t and their history at time $t - n$ (Subfigure 6a) and trending articles (Subfigure 6b). The first figure shows interest drift and the second shows trend-following behavior. Figure 7 shows the ratio of articles that users read that are more similar to their history versus the articles that are more similar to trending articles, showing fractions of personalized reading.

3.6.1 Interest change

Subfigure 6a shows that user interests change over time because history similarity drops by about 25% (from 0.4 to 0.3 or 0.3 to 0.2) over a 14-day period when averaging across all users. The most inactive user group consisting of about a third of all users shows the fastest interest change where they have the highest similarity score to history of 1 day ago, which drops by a third when increasing the time lag. This suggests that history similarity for inactive users

is dominated by short-term interests, and that long-term interests are harder to infer from sparse click embeddings.

More active users have larger similarity scores for both metrics, suggesting that interests of users become more predictable if they are more active. And when comparing the scores, it is clear that average maximum similarity scores are increasingly lower than centroid similarities over increasing user activity, and remain constant over time lag. An explanation for this is that averaging embeddings combines semantic meaning of articles (e.g. "Sports", "Politics" becomes "Sports and Politics"), which can give a more nuanced score for articles that have no close match than averaging maximum similarities. For example, matching "Sports", "Politics" to "Sports, Tech" likely gets a high centroid similarity score because both encode semantics for sports, while averaging the maximum scores gets a lower score of something like 50%.

3.6.2 Trend following

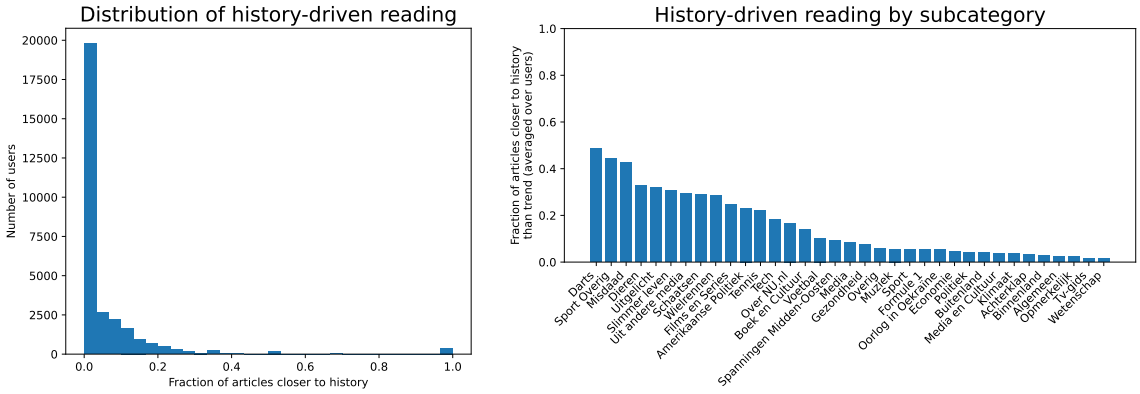
User trend following behavior is visualized in Subfigure 6b. Popular articles are defined by the top N articles per day that together account for 80% of the reads, which on average are 42 articles.

From the average maximum similarity to the popular articles, it is clear that users almost only read popular topics, confirming popularity bias in user behavior, likely caused by a user conformity bias (e.g. a peer pressure) and exposure bias (e.g. visibility of articles on the front page). Centroid similarities don't capture this distinctive trend following behavior as well with more nuanced scores. Interestingly, centroid similarity to the global pool of article reads is higher than similarity to the pool of popular articles, which we think shows that at least some read articles are non-trending, and likely are relevant to a specific set of users, meaning that this is personalized news reading more so than popular news reading. On day 282, we see the largest difference, which was the day that the main election debate was held, and we think that people were reading more than usual diverse personalized topics that they cared about that day.

3.6.3 History-driven reading ratio

Figure 7 further focuses on the exact ratio of read articles that are more similar to the set of popular articles (trend-driven) versus read articles that are more similar to user history (history-driven), averaged across users. High ratios of history-driven reading support the choice for content-driven news recommendation systems.

Subfigure 7a confirms that article reads are mostly trend-driven: most article reads are semantically more similar to the trend than the user history. The history-driven reading ratios over news categories from Subfigure 7b show a more detailed view of the news categories that are better explained by similarity to history than trends. For example, reads in the darts subcategory are almost in 50% of the cases more similar to user history than the current trend,



(a) History-driven user distribution

(b) History-driven reading over categories

Figure 7: Fraction of articles that are more similar to user history than most popular articles, averaged across users, showing that most reads are likely trend-following while some are more history driven.

suggesting that content-based news recommendation systems are effective for recommending articles in this subcategory. Topics like politics, general and economics are more trend-dominant topics when just looking at content, meaning that it is hard to predict their relevance based on history content. So, this motivates interleaving of such articles with recommendations, using additional signal aside from article content such as popularity and collaborative user behavior, and the use of ensembles that can predict confidence of recommendations through model agreement to be able to better handle lower confidence cases like politics through business logic.

3.7 Semantic category similarity versus co-occurrence

The goal of this analysis is to identify shortcomings of semantic embedding similarities in applications in personalized news recommendation. We do this by comparing similarity rankings of *semantic similarity* between article categories with *measured co-occurrence probability* of articles. This measures if similarity prediction by *semantic similarity* between topics A and B captures *interest similarity* worse than SANNR that optimizes for behavior space. Interest similarity represents how frequently users read topics together, which is not equal to semantic similarity.

For example, embeddings for politics and domestics are dissimilar while readers on nu.nl frequently are interested in both. So, in these cross-topic scenarios, semantic similarity should fall short when SANNR implements task adaptation by modeling collaborative behavioral dependencies.

We picked category granularity because we think that it's more interpretable

than fine-grained article-level alignment, and because there are more examples for less noisy probability measurements. We measure article category co-occurrence in user click logs using normalized pairwise mutual information (nPMI), defined as:

$$\texttt{nPMI}(x, y) = \frac{\texttt{PMI}(x, y)}{-\log P(x, y)} \quad \text{, where } \texttt{PMI}(x, y) = \log \frac{P(x, y)}{P(x)P(y)}$$

nPMI ranges from -1 (perfect mutual exclusivity) to 1 (perfect co-occurrence), with 0 indicating independence. We picked nPMI because it normalizes for random co-occurrence, and scales to a fixed range.

We measure the predicted (predicted by the semantic ranker) news category similarity between topics A and B by treating all articles from news category A as user history, and measuring the average predicted similarity to article embeddings from news category B. This creates a directional score that measures if news embeddings from category B are on average relevant for reading histories with the articles from topic A. Because we don't have user IDs in this experiment, we only test the late fusion semantic baseline, SANNR (u) and SANNR (i+u). By iterating over all category pairs, we obtain ranked lists where the first items are predicted as the most similar categories, and the last items the least. We create the same ranked list using measured `nPMI` category co-occurrence probability. The ranked lists allow us to compare the probability metric with the distance metric using Spearman's rank correlations between the lists: $\rho = 1 - \frac{6 \sum_{i=0}^{n-1} d_i^2}{n(n^2-1)}$, where n is the ranked list length, and d_i the rank difference between the assigned rank at position i.

Figure 8 scatters the rank of measured category co-occurrence with nPMI, against the semantical similarity rank of the category embedding centroids. No correlation between the two similarity metrics is present, indicating that BERT can capture no cross-category relevance patterns. Semantic embeddings are not capable of generalizing to other topics than were observed, which is logically a larger problem for more cold-start users, which are most users on the platform.

Table 3 shows examples of the most and least similarly ranked category pairs, measured by semantic content similarity and nPMI co-occurrence probability. Overall, we see that high co-occurrence probability (low nPMI ranks) correspond well to high semantic similarity (low semantic ranks) with some of the smallest rank differences. The largest rank differences are paired with low co-occurrence probability. This exposes a failure mode of semantic similarity where it systematically overestimates similarity for categories that are read together least frequently.

Specific examples from the table show more specific failure modes. Domestics (binnenland) and abroad (buitenland) articles are semantically the most similar subcategories. This makes sense as the content likely covers similar topics, but

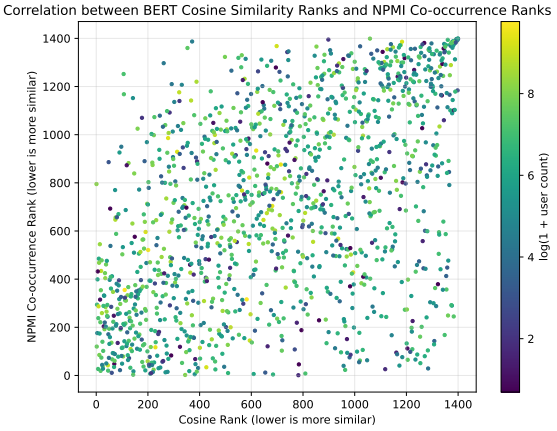


Figure 8: Embedding cosine similarity ranking of category pairs versus measured co-occurrence probability. The Spearman rank correlation of the two axis is 0.295. There is no correlation visible between these similarity scores, indicating that semantic similarity does not capture any measured co-occurrence patterns.

not for recommendation, because the articles talk about fundamentally different regions which is a different type of interest, which is confirmed by the lower nPMI rank. Another equivalent example is soccer (voetbal) and tennis, that are both semantically similar because the articles both cover sports, while they are indicative of different interests. The biking (wielrennen) and domestics subcategories are linked to the largest rank difference. Here, again, semantic similarity overestimates how similar the topics are compared to how often users interact with both. The cases where semantic similarity underestimates behavioral similarity are mostly about connecting sports, media with movies, and completely unrelated topics such as war with movies.

Interestingly, these findings confirm a failure mode that explains some commonly reported issues with the content-based recommender system at DPG Media. Some users for instance reported soccer recommendations, which the table shows can be linked to biking articles. The same commonly happens for regional news, and articles about formula 1.

Note that nPMI probabilities at first seem promising for building a recommender. However, we found that these measurements don't generalize to unseen time steps, and are possibly not fine-grained enough to improve recommendation beyond matching the closest candidate embeddings.

Table 3: Rank disagreement between the semantic subcategory similarity ranking and history co-occurrence probability (nPMI) ranking. Each section shows the five highest and the five lowest ranked pairs under one of two metrics, joined with the other metric's rank and $|\Delta|$ rank gap. Low co-occurrence probability overall corresponds to the highest mismatch with semantic similarity.

Subcategory pair	Semantic similarity	nPMI	$\|\Delta\|$
Ranked by semantic similarity			
Buitenland, Binnenland	1	51	50
Achterklap, Films en Series	2	18	16
Achterklap, Media	3	10	7
Voetbal, Tennis	4	110	106
Wielrennen, Darts	5	4	1
...	...		
Amerikaanse Politiek, Films en Series	347	232	115
Tennis, Oorlog in Oekraïne	348	325	23
Spanningen Midden-Oosten, Formule 1	349	266	83
Oorlog in Oekraïne, Films en Series	350	286	64
Ranked by history co-occurrence (nPMI)			
Darts, Tennis	10	1	9
Darts, Schaatsen	19	2	17
Wielrennen, Tennis	9	3	6
Wielrennen, Darts	5	4	1
Media, Films en Series	15	5	10
...		...	
Tech, Voetbal	202	347	145
Tech, Achterklap	161	348	187
Wielrennen, Binnenland	73	349	276
Tennis, Binnenland	165	350	185

3.8 Discussion

In summary, this analysis sheds light on the following challenges with personalized news recommendation and building NNR systems:

- Most users in MIND and NUMEND have few reads, highlighting a user cold-start; Measured by clicks per user, MIND is more sparse, but measured by clicks per article, NUMEND is more sparse;
- The article lengths and title lengths are more consistent in NUMEND than MIND, which could possibly make sentence embedders and task adaptation for platform article content more effective on NUMEND;
- Survival time of articles is short (and interaction frequencies are skewed) for both NUMEND and MIND, showcasing item cold-start; However, in NUMEND, the survival time is longer, suggesting that more warm items

may be relevant in NUMEND; However, we found that the majority of reads are still within a day of publication;

- MIND does not have a global popularity bias in the impression logs; NUMEND intentionally includes popularity bias in the impression logs in order to approximate the article exposure distribution, and enable models to optimize for user interests instead of article popularity by punishing it for relying on popularity;
- User interests in NUMEND shift over days, and faster for inactive users;
- Long-term interests of less active users in NUMEND are more difficult to capture than more active users, while capturing short-term interests seems to be equally challenging across the user groups;
- All users on average mostly read trending topics;
- Interests of active users in NUMEND are more similar to popular interests *and* active users read more similar to their own history; This could be an artifact of the small candidate pool on nu.nl meaning that users converge to the exposure distribution which is likely close to the popularity distribution on nu.nl;
- The amount of trend following reading fluctuates throughout days in NUMEND;
- Users read more history-driven articles for topics like 'health' than 'general' where users read mostly trend-driven;
- Semantic similarity cannot capture cross-category interests in NUMEND.

Specifically for parameter-free content-based recommender systems that rely on semantic similarity, we learned that it lacks cross-topic generalization and can place dissimilar-interest articles very close. Additionally, this analysis suggests that most user interest is trend-driven, which content-based recommenders cannot model without including a collaborative signal. However, despite the limitations, we think that content-based recommender systems are well-suited in modeling user interests for some news categories. We for instance see that it is effective for some sports categories. Moreover, results show that content-based recommenders are essential for personalized news recommendation to overcome item cold-start, and that short-term interests may be well-modeled with content-similarity.

The ideal NRS approach should:

1. Update (part of) the user profiles frequently (multiple times per day or every time reading history changes) to be able to do short-term recommendations, especially for the least active users and trend-driven topics, possibly with active learning.
2. Not solely rely on article content embeddings from PLM, but perform some level of task adaptation by for example optimizing over impression logs to

be able to recommend unseen topics, include reading trends, and prevent recommending dissimilar interests that are only semantically similar.

This motivates a NRS architecture that has a content-driven base using for instance semantic article embeddings, but also optimizes for collaborative signal to address some limitations with PLM embeddings such as overestimation of topic similarity. Furthermore, frequently updating user representations with new interactions helps to make user embeddings more indicative of user interests, motivating two-tower NNR that can efficiently run inference and training cycles as needed to improve recommender performance.

4 Methodology

This section introduces our method Semantically Aligning Neural News Recommenders (SANNR). Its focus is aligning frozen sentence-encoder embeddings of articles with the user behavioral space, without fine-tuning the encoder. Components from NNR literature, sequential recommender literature, and custom designed ones are combined to handle static embedding input. SANNR is highly modular, allowing for ablation tests to observe which components are effective for this task. SANNR is mainly compared against a semantic late fusion baseline model because SANNR extends on this semantic baseline, because the computational requirements of inference are in the same ballpark[3], and because it is the in-production recommender at DPG Media which SANNR mainly aims to improve. However, because this semantic baseline has not been optimized for recommendation, SANNR is also compared against naive NRMS (Wu et al. 2019c), NAML (Wu et al. 2019a), LSTUR (An et al. 2019), and MINER (Li et al. 2022) implementations with frozen news encoders.

4.1 Article encoding

The baseline recommenders and SANNR that we define in the next sections both build on a frozen news modeler that can encode news articles to meaningful numerical representations. We use sentence encoders for this task because DPG Media uses this in production, and because they encode the textual article data to dense embedding representations where semantic meaning is represented in the direction, which are valuable features for recommendation (Al-Hassan et al. 2024). By default, we use the DistilUSE multilingual sentence encoder. The encoding strategy that we use is also the same as the in-production system at DPG Media: we concatenate each news title with its content, split it into sentences, encode the sentences individually, and average and L2 length normalize all sentence embeddings to get the corresponding news embedding. This strategy is called Sentence-Level (SL) encoding of news title and content (Ortakci et al. 2025). For MIND, news content is the news abstract, and for NUMEND,

[3]Training of SANNR and naive baselines costs substantially more compute than the untrained semantic baseline, but inference time and compute are more important criteria.

news content is the news body.

4.2 Baseline recommenders

The in-production baseline recommender uses cached semantic embeddings encoded by the semantic content encoder to recommend semantically similar articles to user history. This baseline implements item modeling with the DistilUSE sentence encoder with sentence-level encoding. User modeling takes the average of user history article embeddings with vector unit length normalization. Candidate article relevance is the angle between the user embedding and the item embedding, which we compute with dot products over L2 unit length normalized embeddings because this is equivalent to cosine similarity. This recommender is extremely simple and efficient because user history centroids can be computed on-demand, the article content embeddings remain static and can be cached and are not trained on examples. This type of user embedding scoring against candidate article embeddings using average angle is referred to as 'late fusion', because the embeddings are only fused just before scoring (Iana et al. 2023).

NNR from the literature with frozen sentence encoders extend on this baseline with NNR user modeling components and learned fusion mechanisms. Apart from architectural contributions, SANNR implements equivalent NNR components, meaning that frozen NAML, MINER, LSTUR, and NRMS baselines are implemented with SANNR hyper parameters, as defined in Section 4.3.8.

4.3 SANNR

Similar to the baseline, SANNR uses pre-trained sentence encoder(s) to encode textual content of articles to semantic embeddings. These embeddings are part of the model input; the sentence encoder is not optimized, making SANNR encoder-agnostic and efficient. Unlike the baseline, SANNR implements item and user modeling components that are optimized over examples of the downstream recommendation task, allowing for domain adaptation from semantic space to user behavior space. Item-level modeling allows for incorporation of item-level behavioral signals, and user-level modeling allows for incorporation of user-level behavioral signals. However, cross-level signal capture is possible in both directions, but with constraints. User-level optimization has indirect access to item-level signals because candidates cannot be adapted, whereas item-level optimization observes single item embeddings and can only indirectly optimize user representations. The frozen NAML and frozen MINER baselines are implemented with SANNR user modeling components.

Problem statement The key challenge that the methodology of SANNR addresses is the change in the bias/variance balance when optimizing for recommendation with static semantic embeddings as input instead of token-level article content (which is the same as keeping news encoders frozen). Because each article is now a single fixed vector, the model always sees the exact same in-

Table 4: High-level description of all SANNR components and the literature that they are based on.

Module	**Description**
News modeling:	
Item-level adaptation DropoutNet (Volkovs et al. 2017) Delta proj. (Pembek et al. 2025)	Transform the item embedding space to approximate the behavior space.
User modeling:	
User-level adaptation DropoutNet (Volkovs et al. 2017) Delta proj. (Pembek et al. 2025)	Transform the user-level embedding space only, moving user embeddings to more behavioral-representative positions in the semantic space.
User ID embedding LSTUR (An et al. 2019)	Prepend a learned free embedding per user encoding long-term preference, acting as a correction by biasing user representations to individual preference embeddings.
Encode user history NRMS (Wu et al. 2019c)	Encode user history with self-attention to model co-dependence between article reads, an important property of behavior space.
Fuse user history NAML (Wu et al. 2019a) MINER (Li et al. 2022)	Combine user history with additive attention, weighting embeddings more important for representing behavior space higher.

put per item, and token-level regularization such as word masking or dropout is not possible. This makes items easy to identify, so the model can 'remember' individual examples and fit behavior bias such as the exposure bias in NUMEND, instead of learning a semantic matching signal that generalizes to unseen items. SANNR reduces this overfitting with residual connections, implicit and explicit model regularization, and more sampled negatives per positive example, which the methodology further describes in more detail.

Table 4 gives a high-level description of all SANNR components. All components are based on recent developments in recommender literature. Each component is hand-picked, and/or modified to address the limitations of a coarse input space. The news modeling component implements light-weight domain adaptation of static embeddings at news-level. We made this selection of user modeling components to cover a comprehensive range of user modeling strategies

from two-tower neural news recommenders literature. Furthermore, we ensure that each component can be independently included (or excluded) from the SANNR architecture, allowing for detailed ablation analysis of domain adaptation in each component or a combination of components. When all components are disabled, SANNR is equivalent to the late fusion baseline recommender.

The notation that we use throughout this section is as follows:

- $d \in \mathbb{Z}^+$: semantic embedding dimensionality
- $d_{id} \in \mathbb{Z}^+$: ID embedding dimensionality
- $e_i \in \mathbb{R}^d$: semantic embedding of article i
- $e_i^{id} \in \mathbb{R}^{d_{id}}$: item representation of article i (ID embedding)
- $u_l^{id} \in \mathbb{R}^{d_{id}}$: user representation of user l (ID embedding)
- $\mathbf{H}^u = [h_1, \ldots, h_M] \in \mathbb{R}^{M \times d}$: user history embedding matrix with $M \in \mathbb{Z}^+$ items.
- $c_j \in \mathbb{R}^d$: candidate article embedding
- $\mathbf{u} \in \mathbb{R}^{K \times d}$: $K \in \mathbb{Z}^+$ user representation(s)

In the following subsections, we further describe and motivate the methodology of each component, SANNR inference, and SANNR training. We finish with a summarization of the SANNR configurations that together implement equivalent user modeling components to NRMS (Wu et al. 2019c), NAML (Wu et al. 2019a), LSTUR (An et al. 2019), and MINER (Li et al. 2022).

4.3.1 Item-level adaptation

Item-level embedding adaptation optimizes the semantic space of news embeddings for recommendation while maintaining important parts of the semantic space that help with generalization to unseen news. We implement this with delta projections as proposed by Pembek et al. (2025), where each learned collaborative embedding projection adds a small delta to the original semantic embedding. Aside from optimizing embeddings directly for recommendation, we think embedding projection aids down-stream SANNR components by modifying embeddings in arbitrary ways that allow for more stable down-stream gradient descent.

Adapter layers are implemented as follows:

$$f_{\theta,\phi}(e_i) \rightarrow e_i + \theta g_\phi(e_i) \quad \text{and} \quad g_\phi(e_i) = \text{GELU}(\phi \text{LN}(e_i))$$

Here f is the residual adapter function, $\theta \in \mathbb{R}$ is a learned scaler that defines how much of the embedding projection is mixed in, and $g_\phi(e_i)$ is a feedforward neural network that includes parameters $\phi \in \mathbb{R}^{n \times n}$ that project item embeddings. GELU is the activation function of this layer, smoothing the gradient

landscape which can help to improve stability (Hendrycks et al. 2016), and LN (layer normalization) normalizes embeddings over all layer inputs of a single forward pass using the mean and standard deviation, with the goal of normalizing embeddings before the projection is computed (Ba et al. 2016).

We manually tested projection with one, two and three layers, and found that two layers (two learned projections) is a consistently stable trade-off between model capacity and model generalization. Here we have two sets of parameters $\phi_{0,1} \in \mathbb{R}^{n \times n}$. This is defined by:

$$f_{\theta,\phi_0,\phi_1}(e_i) \rightarrow e_i + \theta g_{\phi_1}(g_{\phi_0}(e_i))$$

This is a slight modification from the $f_{\theta,\phi}$ function definition where we do two projections with the g_ϕ function, with independent projection parameters ϕ_0 and ϕ_1. The residual scaling weight θ now scales the double-projected residual.

4.3.2 User-level adaptation

Similar to item-level adaptation, user-level adaptation optimizes news embeddings for recommendation using delta projections. However, dissimilar to item-level adaptation, only user history embeddings are projected, leaving candidate news embeddings as-is. Doing so, user-level adaptation first of all does not invalidate news embedding caches (allowing for more straightforward implementation in practical settings). Second, it optimizes history embeddings in a more restricted way because projected embeddings are still scored against candidates that are in the original semantic space. Additionally, like other user-level components, this component can model directional relevance that item-level adaptation cannot capture. For example, people interested in Formula 1 could be more frequently interested in soccer than the other way around.

Formally, the user-level adaptation implementation is identical to item-level adaptation, but over user history embeddings $\mathbf{H}^u$, and with a new set of weights: θ_u, $\phi_{u,0}$, and $\phi_{u,1}$.

4.3.3 User ID embedding

Inspired by LSTUR (An et al. 2019), this component concatenates the user history with a freely learned user ID embedding. These user ID embeddings use the random user IDs as mappings to an embedding table, that includes the user-specific parameters that are optimized per user. In LSTUR, they propose to use this user embedding as initial hidden state of a gated recurrent unit (GRU), and suggest that these ID embeddings model long-term user preferences. We think that this embedding captures an arbitrary user-specific 'bias' embedding that steers corresponding user embeddings in a good direction for recommendation. For instance, we think that in combination with down-stream GRU or self-attention encoding, such ID embeddings can help resolve ambiguity in the underlying interests of users within their reading history. Think for example of

a reading history of tech articles, while the user might just be interested in the stock prices aspect.

SANNR implements user ID embeddings in an architecture-agnostic way, even allowing for SANNR variants to include user ID embeddings but no encoder such as GRU to learn a combination with user history, or vice versa. This novel modular design allows for ablating ID embeddings and encoder components individually to find the best architecture that balances model capacity with regularization (bias versus variance), which is crucial given the low granularity of input features.

Given the article embeddings of a user history $[e_1, e_2, \cdots, e_k]$, of length $k \in \mathbb{Z}^+$, and a user ID embedding e_{user}, where all embeddings are n dimensional, we concatenate the user ID embedding to the user history to get $[e_{user}, e_1, e_2, \cdots, e_k]$. User ID embeddings are initialized with zero values to ensure they by default do not modify the angle of user embeddings, acting as an unseen user ID.

To improve model generalization to unseen users with no trained user ID embeddings, we apply heavy embedding-level dropout to the user ID embeddings during training, forcing the model to rely on history-based signals rather than memorizing user representations. This approach was first proposed in the DropoutNet cold-start-focused recommender (Volkovs et al. 2017).

4.3.4 Encoding user history

Encoding of user histories allows SANNR to leverage inter-history dependencies to resolve ambiguity in user interests, and increase specificity. An example of such a dependency is a user history containing articles about "ASML", "smartphones", and "Windows 10 tips&tricks", suggesting that the user is likely interested in technology aspects of the articles, and not in stock prices, which could also be an aspect of the ASML topic.

Inspired by user modeling from NRMS, SANNR implements self-attention to encode user histories. However, SANNR uniquely implements a number of modifications to self-attention aimed at improving regularization for stable optimization over lower input granularity:

1. Inspired by (Lu 2020), the key-value projection parameters are shared;
2. In order to enforce use of the semantic space in query, key and value projections, SANNR implements identity matrices (for ablating the completely parameter-free self-attention), or re-uses the previously defined delta projections as query, key, and value projectors;

So, for each query embedding $\mathbf{e}_q$, and the attention target embeddings $\mathbf{e}_{target}$, we define projections to map the embeddings to attention matrices $\{q, k_i, v_i\} \in \mathbb{R}^{3\times d}$ as follows:

$$q = e_q W_q \quad \text{and} \quad k_i = v_i = e_{i,target} W_{kv}$$

Re-using delta projection functions as query, key, and value projectors for self-attention is defined as:

$$q = f_{\theta_q,\phi_q}(e_q) \quad \text{and} \quad k_i = v_i = f_{\theta_{kv},\phi_{kv}}(e_{i,target})$$

In the case of identity matrices, we define these matrices as: $q = k_i = v_i = I$, where I represents the identity matrix.

Using q, k, v, the attention weights α_i are computed using scaled softmax, as proposed by (Lu 2020):

$$\alpha_i = \frac{\exp((qk_i^T)/d)}{\sum_{m=1}^{M} \exp\left(q(e_{m,target}W_{kv})^T/d\right)}$$

$$\alpha = \text{softmax}\left(\frac{q\left(E_{\text{target}}W_{kv}\right)^\top}{d}\right)$$

And finally, we encode each history embedding by 'augmenting' it with neighboring embeddings by taking the weighted sum of the neighboring embeddings using the attention weights that we computed by attending over the neighbors with the query:

$$att(e_q) = \sum_{i=0}^{M} \alpha_m v_m$$

4.3.5 Fuse user history

User history fusion is implemented with additive attention, similar to the final learned fusion layers of the user encoders in NAML and MINER. In simple terms, additive attention learns mappings from embeddings to weights for a weighted sum that aggregates history embeddings into one or a few user embeddings. This type of fusion is referred to as *early fusion*, because the user embedding is aggregated before scoring. Late fusion may preserve the full user history until scoring because it simply averages the embeddings uniformly, which is cheap enough for most inference pipelines.

The goal of early fusion is to learn which user history embeddings are important in recommending relevant articles. So for example, when a user reads skating and politics articles, additive attention could learn to weigh skating more because it is more niche and could better represent distinct user interests. We also implement MINER's poly-attention scheme, which is a variant where *multiple* user embeddings are learned from their history, with the goal of capturing multiple aspects of user interests. Users might for example read articles about Google because they're a fan of tech *and* are interested in the tech stock market. Even though MINER is designed to learn to capture these aspects jointly with fine-grained token-level news encoders, we think the embedding space might be

ambiguous enough to allow for multiple aspects of user interests to be extracted, and could furthermore go hand-in-hand with other SANNR components.

Additive attention from Wu et al. (2019a) optimizes a context code that attends over the user history to find a weight for each item. These weights are used to combine the items in a weighted sum. Poly attention from Li et al. (2022) does the same, but optimizes a set of codes, and creates a set of weighted sum $K \in \mathbb{Z}^+$ user representations. This is achieved by optimizing K context codes $\mathbf{c} = \{c_1, c_2, \cdots, c_K\}$, with code dimensionality of D, where $c \in \mathbb{R}^{K \times D}$, $D \in \mathbb{Z}^+$ that attend over user history $\mathbf{H}^u$. Given the codes, user embeddings are extracted as follows:

$$u_i = \sum_{j=1}^{M} \text{softmax}(c_i^T \tanh(\mathbf{W}^h h_j)) \cdot h_j$$

Where $\mathbf{W}^h$ is a globally learnable matrix of parameters that maps h_j to the dimensionality of the codes, adding some non-linearity in the process, but mainly allowing for a lower dimensionality for the freely learned code vectors so that they overfit less easily. This represents the weights of a fully connected feed-forward layer without bias. This poly attention definition is simplified from the MINER paper (Li et al. 2022) to not include candidate category embeddings as this only slightly increases recommender performance for a large increase in architectural complexity.

Now, the user representation consists of K embeddings, or M embeddings if no poly attention was applied. Before scoring, we uniformly average these user representations, and apply L2 normalization. SANNR adopts averaging because it is most simple to implement while achieving the best performance in MINER (Li et al. 2022).

4.3.6 Scoring

The scoring function takes candidate items and user representations, and assigns a relevance score to each item–user pair, estimated with dot products. Specifically, for each user, all their user representations are averaged, L2 normalized, and matched against all candidate article representations using a dot product (because this is equivalent to cosine similarity for L2 normalized embeddings). In poly attention user representations, this implements averaging pooling, which worked best in MINER for MIND (Li et al. 2022). When user history fusion is disabled, this implements late fusion.

4.3.7 Model training

Training SANNR is efficient because there are few learnable components, so we train with a relatively large batch size of 256, which still requires less than 2GB

of graphics memory. We train SANNR using 20 epochs, with early stopping patience of 5, targeting validation NDCG@10 ranking performance.

Similar to NRMS (Wu et al. 2019c), SANNR employs Noise Contrastive Estimation (NCE) loss that simultaneously optimizes for positive and negative examples within a list of candidates. During training, the first candidate is always positive, and the next 4 are uniform randomly sampled from the impression targets. This allows for implementation of NCE loss using Binary Cross Entropy (BCE) loss, which is part of the pytorch library.

We implement tailored engineering solutions to train SANNR without overfitting on the training distribution, and the popularity distribution. In summary, these solutions are:

- Adding input noise to embeddings;
- Randomly dropping history embeddings;
- In-batch negatives;
- Time-invariant contrastive loss;
- Increased learning rate of ID embeddings.

Input noise Adding noise to history embeddings is a form of data augmentation, which is commonly used to reduce overfitting by increasing the amount of input samples (Miyato et al. 2017). Adding noise to embeddings is common in the related domain of LLMs (Jain et al. 2023). They define this as:

$$\mathbf{H}^u_{noise} = \mathbf{H}^u + \frac{\tau}{\sqrt{dM}}\mathbf{N} \quad \mathbf{N} \sim \mathcal{U}(-1, 1)$$

Where $\mathbf{H}^u_{noise}$ are the user history embeddings with noise, $\tau \in \mathbb{Z}^+$ is a tunable parameter we set at 5, and N is a noise vector that's sampled from a uniform random distribution with values between -1 and 1.

History embedding dropout Randomly dropping history embeddings during training replicates the chance that users accidentally clicked on some irrelevant article (click log noise in implicit feedback) during training. Doing so, we ensure that model optimization is more resilient to noise in the dataset, reducing variance. In addition, this method prevents the model from memorizing specific article combinations, thus forcing it to generalize more.

In-batch negatives In-batch negatives are drawn from in-batch positives of other instances within the same training batch. Because these negatives are sampled based on their frequency in the training batch, in-batch negatives implicitly sample from the item popularity distribution. This (i) acts as a popularity penalty, forcing the model to learn user interests instead of popularity. (ii) Second, this makes for a much more difficult task of distinguishing *one* positive

Table 5: SANNR component combinations that implement equivalent user modeling architectures to the established neural news recommenders: MINER (Li et al. 2022), NAML (Wu et al. 2019a), NRMS (Wu et al. 2019c), and LSTUR (An et al. 2019). Adapt (i) and Adapt (u) refer to the item adapter and user adapter component respectively, +ID embeddings refers to the learned user ID embedding, enc. refers to encoding of user history and fusion refers to user history fusion using additive or poly-attention.

User modeling	**Adapt (i)**	**Adapt (u)**	**+ID embs.**	**Enc.**	**Fusion**
MINER	○	○	○	○	poly-att.
NAML	○	○	○	○	add. att.
NRMS	○	○	○	●	add. att.
LSTUR	○	○	●	●	add. att.

example from 256 *more* negative examples than before, which reduces variance in the gradient and makes for a stronger gradient.

Time-invariant contrastive loss Time-invariant contrastive loss, inspired by Zheng et al. (2021), ensures that the created user embeddings are a bad predictor of the day of year of relevant article reads. In other words: user embeddings may not predict candidate item relevance merely based on the time stamp of the example; it has to learn time-invariant, non-trending representations that generalize to unseen days.

ID embedding learning rate Lastly, we increase the learning rate of the ID embeddings in steps of $\times 10$. This allows the ID embeddings to adopt the user-, or item-level signal faster, enabling other model parameters to learn cross-user, cross-item patterns that generalize to unseen users or items. Another reason for increasing the learning rate is that these embeddings are learned freely from scratch, unlike projections that should make smaller steps because they *scale* valuable input embeddings.

4.3.8 Connection to user modeling literature

Table 5 connects SANNR component combinations to neural news modeling literature. SANNR can be configured to implement user modeling from NAML and NRMS. Additionally, SANNR can implement LSTUR and MINER in an *equivalent* way: in LSTUR user modeling implementation, we replace the GRU layer with self-attention and additive attention; and, we don't consider candidate category embeddings in the MINER implementation. LSTUR originally implements user ID embeddings as the initial hidden state in a GRU layer that processes the user history sequence to one embedding instead, but we argue that self-attention with additive attention approximates performance of this approach because it can also model sequential dependencies and learned history

Table 6: Four SANNR variants, ordered by increasing practical implementation difficulty. All variants allow for any user modeling optimization using any user-level SANNR component.

	Adapt (i)	**Any user modeling**	**+ID embs.**
1. SANNR (u)	○	●	○
2. SANNR (ID+u)	○	●	●
3. SANNR (i+u)	●	●	○
4. SANNR (i+ID+u)	●	●	●

aggregation. Our MINER implementation is not architecturally different, but just one of the simplifications that the authors propose.

4.3.9 Key SANNR variants

Among all the architectures that we can implement by toggling and combining SANNR components, we distinguish between four variants that are increasingly difficult to practically implement, listed in Table 6. Item adaptation is inefficient because it invalidates caches, and user ID embeddings are expensive to maintain as new users sign up. We think the item adaptation is a larger hassle practically, so we ranked those variants last.

5 Experiments

We design experiments to test the methodology, and answer the research questions that we introduced. Experiments are sectioned by research question.

5.1 Semantic baseline

How do article encoding strategies (such as which text to use and how they are pooled) and the sentence encoder family affect zero-shot personalized news recommendation using the semantic embeddings without task optimization steps?

5.1.1 Experimental setup

To test which sentence encoders and encoding strategies are most effective for zero-shot news recommendation, we cover a number of encoder families and strategies in a grid search. This of course includes the DistilUSE sentence encoder with the sentence-level title+content encoder that we defined in the methodology, but also some additional encoder families and strategies. We

Table 7: Summary of Sentence Transformer models per category that they are optimized for: all-round common use, retrieval, and semantic text similarity (STS). The name of each model is listed, joined by the parameter count and embedding dimensionality.

Category	Model	Type	Params	Dimension
Common	DistilUSE	Encoder	135M	512
	MPNet-multi	Encoder	278M	768
	MiniLM-L6	Encoder	22M	384
	BERTje	Encoder	109M	768
Retriever	E5-large-multi	Encoder	560M	1024
	BGE-M3	Encoder	568M	1024
	Qwen3-Embed	Decoder	600M	1024
STS	Jasper-TC	Decoder	600M	1024
	B1ade-embed	Encoder	137M	1024
	B1ade-embed-KD	Encoder	33M	1024
	Mini-GTE	Encoder	33M	384

start with describing how we select sentence encoder families for this experiment, continue with describing encoding strategies, next describe prompts and prefixes that we test that are recommended for some sentence encoders, and finish with a summary of the two experiments that we conduct.

Encoder families Selecting sentence encoder families for this experiment is not trivial because, to the best of our knowledge, very few benchmarks include recommendation performance of sentence encoders. Massive Text Embedding Benchmark (MTEB) is one of the leading references of pre-trained sentence transformers, and their performance with their public leaderboard. We pick three state-of-the-art Dutch news retrievers optimized for matching relevant passages to queries, and four state-of-the-art Dutch news semantic text similarity classifiers optimized for matching semantically similar texts. So, here we sorted on sentence encoder performance on Dutch datasets. Lastly, we pick some commonly used ones such as DistilUSE which is used at DPG Media. Table 7 shows the picked models.

Strategies Apart from the sentence level title+body encoder strategy, we experiment with only encoding article titles, and with encoding article titles+content with no sentence-level pooling.

Encoder prompts Because the Qwen3-Embed and Jasper-TC models are decoder models that require prompts, we implement the following prompts to test their in-context learning ability:

1. "Identify the high-level category and specific sub-topics of this news article for recommendation" – push to encode topics;
2. "Identify the topic and reader interest category of this news article" – push to encode topic and user interests;
3. "Represent the reading interests and persona of a user who would enjoy this news article" – push to encode user interests and user personas
4. "Retrieve the news article with the exact same meaning" – push to encode meaning of article, increasing specificity (e.g. embedding articles about soccer further from articles about table tennis).

Prefixes The E5-large-multi and BGE-M3 models are fine-tuned for asymmetric comparison (retrieval) where the query is encoded differently than the passages, which is respectively enforced by the 'query: ' and 'passage: ' prefix. In recommendation, the task is symmetrical because we match documents to documents or passages to passages. Therefore, only the 'passage: ' prefix is tested.

Experiments In summary, we conduct two experiments. First, we test prompts for the prompt-based decoder embedders Qwen3-Embed and Jasper-TC on the test splits of the MIND and NUMEND datasets using the nDCG@10 ranking metric. In this first experiment, we use the encoding strategy of concatenating the title with the article content, and passing this full string to the models. We use the best-found prompt for the next part. In the second part, we test all encoding strategies in combination with encoder families, in a grid-like manner, and score the news encoders by nDCG@10 and precision@1 on the MIND and NUMEND test splits. Because sentence encoders are deterministic, we don't average the performance across repetitions.

5.1.2 Results: encoder prompts

Table 8 shows the recommender ranking performance of four prompts for the Qwen3-Embedding-0.6B embedding model as news encoder with late fusion user modeling. From this, it is clear that more simplistic prompts targeting topic representation score slightly better on NUMEND. For MIND, we see much larger differences, with the top prompt scoring 0.410 nDCG@10, which approaches the unfrozen NNR performance that we list in Section 5.2.4. This could indicate that the embedder is most sufficient at English, resulting in better instruction following.

5.1.3 Results: encoder strategies and families

Table 9 shows the ranking and classification performance of the selected sentence encoders used for user modeling in the late fusion recommender. The best top-10 ranking and top-1 classification model on MIND is Qwen3-Embed, and

Table 8: Recommender ranking test performance (nDCG@10) of the late fusion recommender using embeddings encoded by Qwen3-Embedding-0.6B with four prompts from concatenated title and content.

prompt	MIND N@10	NUMEND N@10
Identify the topic and reader interest category of this news article:	0.408	0.533
Retrieve the news article with the exact same meaning:	0.387	**0.540**
Represent the reading interests and persona of a user who would enjoy this news article:	0.369	0.537
Identify the high-level category and specific sub-topics of this news article for recommendation:	**0.410**	0.538

the best on NUMEND is BGE-M3. The optimal strategy differs per model and dataset, but overall the truncated title-with-content encoding performs best for both nDCG@10 and precision@1. BGE-M3 stands out: it is the weakest model on MIND across both metrics, but the strongest on NUMEND. Likely, it trades off performance on English news texts for broader multilingual and Dutch performance. Only MIND clearly benefits from larger sentence encoders such as Qwen3-Embed or Jasper-TC. On NUMEND capacity matters less: smaller sentence encoders such as DistilUSE outperform Qwen3-Embed. We see that multilingual ability is more important for NUMEND than scale: models (such as Qwen3-Embed) that only implement mixed-language pre-training underperform compared to retriever models (such as BGE-M3 and E5-large-multi) that implement additional strategies for retrieving multilingual data.

5.2 Architecture for frozen news encoders

Which lightweight news and user modeling components (and component combinations), drawn from NNR and sequential recommendation literature, add the biggest improvements to recommendation from static semantic user history embeddings as model input? And to what extent do modifications for handling frozen news encoders (such as embedding projection) improve over (i) the parameter-free semantic baseline and (ii) established NNR architectures (e.g. NRMS, LSTUR, MINER, NAML) applied to frozen embeddings?

Table 9: Sentence encoder news modeling recommender performance (nDCG@10, and precision@1) on MIND and NUMEND test splits, grouped by model category and encoding strategy. User modeling is implemented with parameter-free late fusion. Best scores per model are boldfaced, and best scores per dataset are underlined. The models are grouped by task type that they excel at.

		MIND		NUMEND	
Model	Strategy	P@1	N@10	P@1	N@10
Common					
DistilUSE*	Title	0.163	0.367	0.255	0.521
	Title+content	0.175	**0.392**	0.289	0.544
	Title+content (sl)	**0.178**	0.386	**0.291**	**0.548**
MPNet-multi	Title	0.172	0.374	0.247	0.518
	Title+content	**0.179**	**0.388**	**0.275**	**0.535**
	Title+content (sl)	0.173	0.378	0.267	0.532
Retriever					
E5-large-multi	Title (prefix)	0.167	0.374	0.271	0.531
	Title+content (prefix)	**0.179**	**0.386**	**0.295**	**0.550**
	Title+content (prefix,sl)	0.128	0.330	0.261	0.529
BGE-M3	Title (prefix)	0.143	0.339	0.256	0.522
	Title+content (prefix)	**0.175**	**0.375**	**0.301**	**0.555**
	Title+content (prefix,sl)	0.137	0.321	0.252	0.523
Qwen3-Embed	Title (prompt)	**0.194**	**0.408**	0.265	**0.536**
	Title+content (prompt)	0.191	0.405	**0.280**	0.533
	Title+content (prompt,sl)	0.167	0.369	0.264	0.533
STS					
Jasper-TC	Title (prompt)	0.188	0.402	0.250	0.523
	Title+content (prompt)	**0.193**	**0.406**	**0.275**	**0.543**
	Title+content (prompt,sl)	0.162	0.367	0.263	0.532
B1ade-embed	Title	0.188	**0.403**	0.236	0.505
	Title+content	**0.193**	0.402	**0.264**	**0.529**
	Title+content (sl)	0.158	0.356	0.239	0.506
B1ade-embed-KD	Title	**0.184**	**0.390**	0.211	0.484
	Title+content	0.168	0.379	**0.239**	**0.505**
	Title+content (sl)	0.134	0.341	0.230	0.500
Mini-GTE	Title	0.175	0.376	0.210	0.486
	Title+content	**0.180**	**0.377**	**0.244**	**0.512**
	Title+content (sl)	0.160	0.365	0.233	0.502

5.2.1 Experimental setup

We answer the research question in three experiments. First, to find which components from NNR literature are effective for handling frozen news encoders,

we do a large-scale ablation study where all combinations of components are tested on NUMEND and MIND. From this ablation study, we identify the exact configurations for the four SANNR variants that we use for the next two experiments, defined by the criteria previously listed in Table 6. Next, we ablate SANNR modifications: regularization strategies, and projection strategies. Third, we compare the variants to all baselines, including frozen NNR implementations on MIND and NUMEND, external unfrozen NNR performance on MIND, and the semantic baseline. We run each configuration three times to obtain a standard deviation, mainly use the nDCG@10 ranking performance, but also report precision@1 and training and inference time per 1000 samples for the baseline comparison.

Component ablation All SANNR component combinations are tested in a grid search, that also includes additional hyperparameters necessary for optimization such as learning rate and component-specific parameters such as self-attention key-value-projection. Table 10 lists the high-level hyperparameters for toggling all SANNR components and the corresponding component-specific hyperparameters. The components (that are also illustrated in Figure 1) are listed in order of execution. From the grid search, we create a 'flow chart' in the same order that lists the NDCG@10 validation scores and corresponding test scores of the best configuration per component combination, allowing us to interpret how SANNR components interplay, and what configurations are most and least effective for each dataset. This flow chart is especially suited for ablating SANNR components because of its simplistic, sequential architecture, as shown in Figure 1. The configuration per subgroup is picked from this architectural ablation study under two conditions: (i) it has the best validation NDCG@10 on SANNR for that variant, (ii) the gap between the validation and test NDCG@10 is not larger than 0.02 as this indicates poor generalization and would need to be re-trained more frequently. We show the four picked SANNR variants in the graph, and additionally label which SANNR architectures are equivalent to NNR literature.

Regularization and projection ablation Next, we ablate the components that further set SANNR apart from NNR literature: model regularization, embedding projection, and query-key-value projection. We ablate regularization techniques for each SANNR variant in the order of: (i) in-batch negatives, (ii) input noise, (iii) history dropout, and (iv) adversarial loss. Next, we ablate news-level embedding projection for the SANNR (i+u) and SANNR (i+ID+u) variants by testing: (i) naive linear projection, (ii) delta projection, and (iii) no projection. Lastly, we ablate query-key-value projection in the same order.

Baseline comparison The comparison to the baselines includes training and inference compute per 1000 samples to highlight the main purpose of frozen NNR: training/inference efficiency. As baselines, we include the semantic baseline that all frozen NNR and SANNR build upon. Furthermore, we implement

Table 10: Hyperparameter search space over all SANNR components. Disabling every component is equivalent to the late fusion baseline. Each configuration is run 3 times. Global hyperparameters (e.g., learning rate) are omitted.

Component	Variants
Item adapter	(i) semantic embedding projection, (ii) no adaptation
User adapter	(i) semantic embedding projection, (ii) no adaptation
User ID embedding	(i) prepended to history, (ii) disabled
User encoder	*QKV projection:* (i) residual adapter layer, (ii) linear projection, (iii) identity (no projection)
User fusion	(i) late fusion (uniform mean), (ii) additive attention, (iii) poly-attention

the NAML, MINER, NRMS, LSTUR, and NRMS baselines with SANNR hyper parameter settings. NAML is implemented with additive attention ($K = 1$), MINER with poly attention ($K = 4$), NRMS with multi-head self-attention with 16 heads fused with additive attention with $K = 1$, and LSTUR with self-attention and additive attention ($K = 1$) over user history with a user ID embedding. For all frozen NNR implementations, we disable any SANNR regularization, and use linear QKV projections. Furthermore, we compare against external NNR baselines, sourced from the FUM (Qi et al. 2022) and NaSE papers (Iana et al. 2025). Qi et al. (2022) list well-tuned NNR performance on MIND and corresponding training/inference compute. Iana et al. (2025) list performance of the NaSE sentence encoder that is tuned for the news recommendation domain. Even though NaSE is a sentence encoder, we did not include this in the semantic baseline experiment, because it is specifically tuned for the news domain, which SANNR implements through embedding projections, and user modeling.

5.2.2 Results: architecture ablation

Figure 9 shows the ablated SANNR components in a flow chart, and lists the NDCG@10 ranking performance of SANNR using the corresponding component combinations. All configurations achieve better performance than the parameter-free late fusion baseline on NUMEND (> 0.548 validation NDCG@10) and MIND (> 0.364 validation NDCG@10). The best configuration on NUMEND combines item-level embedding adaptation with user ID embeddings and self-attention history encoding, improving the baseline with $0.548 \rightarrow 0.588$ validation NDCG@10. The best configuration on MIND combines user ID embeddings and self-attention encoding, improving the baseline with $0.364 \rightarrow 0.384$ validation NDCG@10. We show the results of the validation performance of SANNR components in Figure 9 in order of execution, which is adapt(i) $\rightarrow$ adapt(u) $\rightarrow$ ID embedding $\rightarrow$ encode $\rightarrow$ fuse. Together with the results of sin-

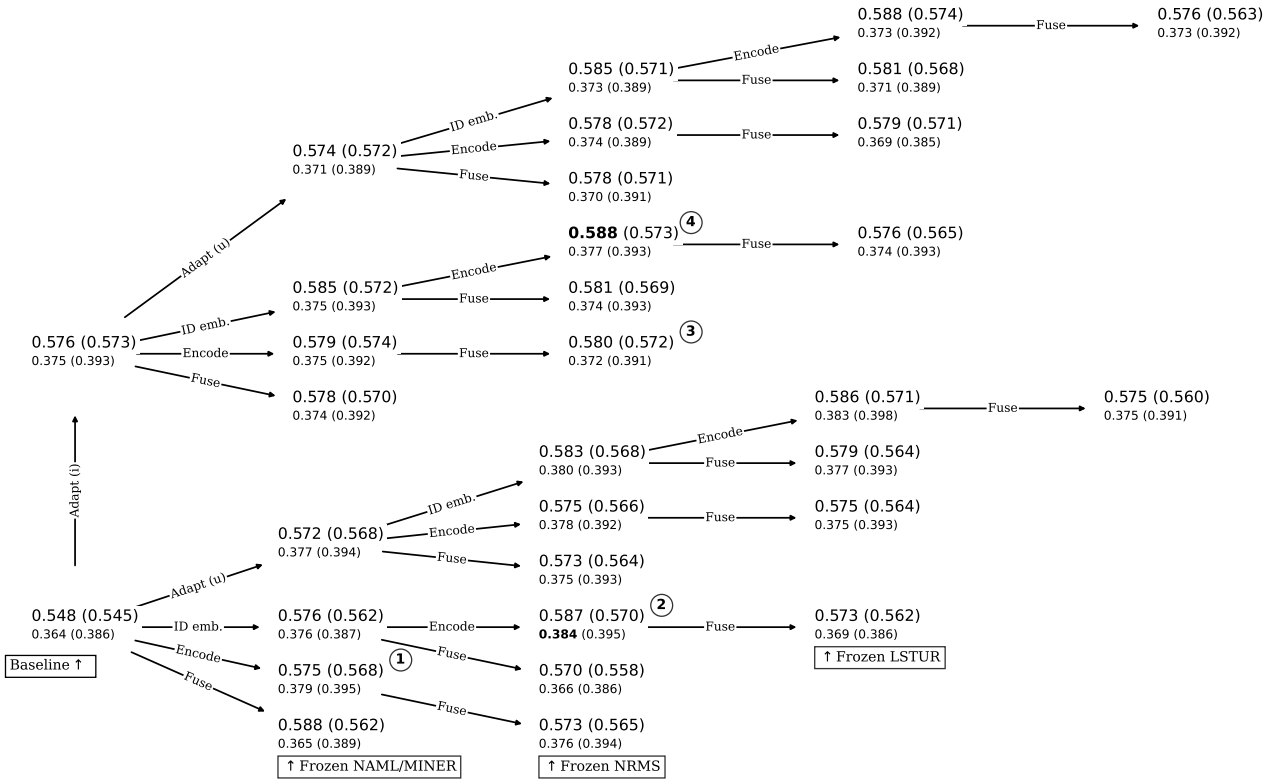


Figure 9: Flow graph ablating the main SANNR components. Each edge represents including a SANNR component, and each node shows the corresponding validation NDCG@10 and bracketed test NDCG@10 when enabling all predecessor edge components. Scores on NUMEND are listed in large text, and scores on MIND on small text. Each configuration is averaged over 3 repetitions. The root node represents the parameter-free late fusion baseline. Boldface indicates the best result for each dataset.

gular components, we interpret how it interplays with any previous components. We finish with an interpretation of validation/test performance gaps that are clear from the figure.

Adapt(i) News embedding adaptation (Adapt(i)) improves over the baseline for both datasets ($0.548 \rightarrow \mathbf{0.576}$ on NUMEND and $0.364 \rightarrow \mathbf{0.375}$ on MIND). This could be the result of it (i) reducing anisotropy (the tendency of semantic embeddings to cluster closely together (Zhang et al. 2024)) by 'pushing' embeddings apart for more discriminative similarity scoring, and/or (ii) encoding

behavioral similarity. It is one of the most effective components at doing so.

Adapt(u) User-level history embedding adaptation (Adapt (u)) is a good alternative to news embedding adaptation (Adapt (i)) because it also improves over the parameter-free baseline (0.548 → **0.572** on NUMEND and 0.364 → **0.377** on MIND) while it does not require additional item-level inference. The user adaptation branch overall performs slightly worse than item-level adaptation on SANNR, but on MIND, user history embedding adaptation outperforms item-level embedding adaptation in most cases, for example when only applying user-level adaptation: 0.377, versus 0.375 validation NDCG@10 when applying item-level adaptation. Item-level adaptation is less expressive than user-level adaptation at capturing behavioral signal because of the constraint of scoring against candidates in the non-projected embedding space, so the performance increase on MIND shows that this dataset benefits from more item-level regularization and the decrease on NUMEND shows that it benefits from more capacity in item adaptation. Linking back to dataset statistics, this makes sense: user-item interactions in MIND are more sparse because it contains many more unique items and about the same amount of interactions.

Adapt (i) → Adapt (u) Combining news embedding adaptation (Adapt(i)) and user history embedding adaptation (Adapt(u)) is not effective (0.548→ 0.576 → 0.574 on NUMEND and 0.364 → 0.375 → 0.371 on MIND), which suggests that they optimize using the same part of the gradient. And because news embedding adaptation is much more expressive than history embedding adaptation, this shows that news embedding adaptation already absorbs most of the gradient for optimizing behavioral item similarity and the embedding space, leaving little room for the less expressive history embedding adapter to optimize. So, combining the two just causes the model to overfit more.

ID embedding Standalone, ID embeddings improve the baseline by 0.548 → 0.575 on NUMEND, and 0.364 → 0.376 on MIND, which even surpasses item-level adaptation performance. This shows that ID embeddings in this scenario capture user preference bias embeddings that are effective for personalized news recommendation in both datasets.

ID embedding → Adapt Stacking user ID embeddings with embedding adaptation components is very effective. For example 0.548 → 0.572 → 0.583 on NUMEND when we stack item-level adaptation with user ID embeddings. This confirms that user ID embeddings capture different aspects of behavioral signal than adapters: likely more user-level behavioral signal in terms of collaborative filtering.

Encoding Encoding user history embeddings using self-attention without any embedding adaptation or ID embeddings improves the baseline by 0.548 → 0.575 on NUMEND, and by 0.364 → 0.379 on MIND. This outperforms the SANNR

configuration with just ID embeddings on MIND, but not on NUMEND, likely because user-level embedding adaptation, which is implemented by self-attention value projection, is more effective on MIND. On NUMEND, item-level adaptation performs slightly better than encoding (0.576 vs 0.575), but encoding outperforms user-level adaptation (0.575 versus 0.572). On MIND, encoding performs slightly better than both user-level and item-level adaptation (0.379 versus 0.377 and 0.375). So, the self-attention mechanism is very effective, even by itself.

ID embedding → Encoding Combining ID embeddings with encoding proves to be specifically effective on the validation splits of both datasets, achieving 0.587 validation NDCG@10 on NUMEND, and 0.384 validation NDCG@10 on MIND, which is the best performing configuration on MIND. Similar to combining ID embeddings with embedding adapters, encoding complements with user-level collaborative-like signal that user ID embeddings capture. Furthermore, because this combination of components implicitly uses the user preference embeddings as attention queries, this suggests that this personalized attention variant is effective, acting as a bias to enrich user history embeddings with user preferences.

Adapters → Encoding Encoding without user-level adaptation achieves slightly better performance than encoding with user-level adaptation on both datasets and splits ($0.548 \rightarrow 0.575$ versus $0.548 \rightarrow 0.572 \rightarrow 0.575$ on NUMEND, and $0.364 \rightarrow 0.379$ versus $0.364 \rightarrow 0.377 \rightarrow 0.378$ on MIND). It is likely sufficient to just model inter-history context to further enrich user history representations instead of first adapting the embeddings. Encoding without item-level adaptation achieves comparable performance to encoding with item-level adaptation on NUMEND, and slightly better performance on MIND. This difference in performance across the datasets again indicates that the SANNR configurations more quickly overfit on MIND than on NUMEND, especially when adapting item embeddings.

Fusion Early fusion, indicated as 'fusion' in the figure, increases the performance in a few, often seemingly random, configurations. The best early fusion configuration implements additive attention over static history embeddings (standalone fusion), improving the baseline with $0.548 \rightarrow 0.588$ on NUMEND, making it one of the best configurations on the validation set. On MIND, the baseline is improved by a relatively small amount: $0.364 \rightarrow 0.365$. This shows that additive attention assigns lower weights to embeddings that are less indicative of user preferences such as generic embedding, and higher weights to more indicative embeddings. However, while standalone fusion performs really well on the validation set of NUMEND, it is one of the worst for generalizing to the test set. This shows that it likely upweights short-term trending interests, which doesn't generalize to three days into the future (the test set). The reduced performance and val/test score difference on MIND either indicates that

trends in MIND shift more quickly, that impression logs are more resilient to this type of overfitting, or that the interaction sparsity in MIND prevents this behavior.

In combination with other components, early fusion slightly increases recommender performance when combined with adapters, such as item-level adaptation with early fusion $0.548 \rightarrow 0.576 \rightarrow 0.578$ on NUMEND. This suggests that these components 'work together' in highlighting important aspects of user preferences. However, in combination with user ID embeddings, early fusion is not effective. This suggests that early fusion likely resorts to upweighting the freely learned user ID embedding to reduce the loss during training, resulting in overfitting on the training set.

Test-validation gap SANNR improves validation performance by much more than test performance, especially when including user ID embeddings. At a first glance this seems like a classical situation where a configuration overfits on the validation set. However, each configuration shown in the figure does not actually include *that many* configurations for it to be able to overfit on the validation set by picking the best hyperparameters. So, also keeping in mind the temporal splits, this shows that these configurations overfit on behavioral patterns that shift within two to three days, which are likely short term interests. User ID embedding validation set overfitting confirms this: it captures user-level interests, and those are known to be especially volatile.

SANNR variants Figure 9 labels the four SANNR variants (labeled 1-4) that are selected for further SANNR analysis. Table 14 shows the NDCG@10 and precision@1 scores on NUMEND and MIND validation and test splits, and corresponding standard deviations, of these four SANNR variants. Standard deviation is low for most configurations, apart from SANNR (i+u) test performance. On MIND, this is not the case because the test set is sampled from the validation set making it temporally closer to the validation set. Even though we did not include statistical significance as we just tested over repetitions, the low standard deviation for other configurations gives a good indication of a consistent improvement over the baseline by all SANNR variants.

5.2.3 Results: regularization and projection ablation

Table 11 shows the results of the ablation experiments of SANNR regularization strategies, Table 12 shows the results for the ablation of embedding projection, and Table 13 ablates query, key, and value self-attention projections.

Regularization ablation Table 11 shows the ablation of regularization strategies proposed in SANNR methodology for the four SANNR variants on MIND and NUMEND. On NUMEND, we see that regularization on SANNR (u) has no effect. However, regularization for all other variants improves ranking performance, meaning that variants with increased model parameters benefit more

Table 11: Ablation of training regularization strategies in SANNR. Performance is measured by val/test nDCG@10, subscripts show standard deviation over 3 repetitions, and bracketed scores show test performance. Batch negatives are most effective, but only on NUMEND (apart from the SANNR (u) variant).

	Full	−batch neg.	− input noise	−hist. dropout	−adv. loss
NUMEND					
1: SANNR (u)	**0.585**$_{\pm 0.000}$ (0.578$_{\pm 0.001}$)	**0.585**$_{\pm 0.001}$ (0.578$_{\pm 0.001}$)	0.584$_{\pm 0.001}$ (0.577$_{\pm 0.001}$)	**0.585**$_{\pm 0.000}$ (0.578$_{\pm 0.001}$)	**0.585**$_{\pm 0.001}$ (0.578$_{\pm 0.001}$)
2: SANNR (ID+u)	**0.593**$_{\pm 0.000}$ (0.578$_{\pm 0.000}$)	0.591$_{\pm 0.000}$ (0.579$_{\pm 0.000}$)	0.591$_{\pm 0.000}$ (0.579$_{\pm 0.000}$)	0.591$_{\pm 0.000}$ (0.577$_{\pm 0.001}$)	0.591$_{\pm 0.000}$ (0.577$_{\pm 0.000}$)
3: SANNR (i+u)	**0.589**$_{\pm 0.001}$ (0.580$_{\pm 0.001}$)	0.584$_{\pm 0.001}$ (0.577$_{\pm 0.001}$)	0.585$_{\pm 0.001}$ (0.578$_{\pm 0.003}$)	0.585$_{\pm 0.001}$ (0.576$_{\pm 0.003}$)	0.583$_{\pm 0.001}$ (0.576$_{\pm 0.003}$)
4: SANNR (i+ID+u)	**0.596**$_{\pm 0.001}$ (0.582$_{\pm 0.001}$)	0.594$_{\pm 0.001}$ (0.579$_{\pm 0.001}$)	0.594$_{\pm 0.000}$ (0.579$_{\pm 0.001}$)	0.594$_{\pm 0.000}$ (0.578$_{\pm 0.000}$)	0.593$_{\pm 0.000}$ (0.578$_{\pm 0.002}$)
MIND					
1: SANNR (u)	0.378$_{\pm 0.001}$ (0.395$_{\pm 0.000}$)	0.380$_{\pm 0.001}$ (0.395$_{\pm 0.003}$)	0.380$_{\pm 0.000}$ (0.393$_{\pm 0.004}$)	**0.381**$_{\pm 0.000}$ (0.395$_{\pm 0.001}$)	**0.381**$_{\pm 0.001}$ (0.392$_{\pm 0.001}$)
2: SANNR (ID+u)	0.382$_{\pm 0.001}$ (0.393$_{\pm 0.000}$)	0.383$_{\pm 0.000}$ (0.394$_{\pm 0.000}$)	0.383$_{\pm 0.001}$ (0.394$_{\pm 0.000}$)	0.383$_{\pm 0.001}$ (0.393$_{\pm 0.001}$)	**0.386**$_{\pm 0.000}$ (0.391$_{\pm 0.000}$)
3: SANNR (i+u)	0.366$_{\pm 0.003}$ (0.386$_{\pm 0.003}$)	0.367$_{\pm 0.002}$ (0.386$_{\pm 0.004}$)	0.368$_{\pm 0.002}$ (0.385$_{\pm 0.003}$)	**0.369**$_{\pm 0.002}$ (0.387$_{\pm 0.004}$)	0.368$_{\pm 0.002}$ (0.388$_{\pm 0.007}$)
4: SANNR (i+ID+u)	0.372$_{\pm 0.000}$ (0.387$_{\pm 0.001}$)	**0.382**$_{\pm 0.001}$ (0.391$_{\pm 0.002}$)	0.378$_{\pm 0.001}$ (0.392$_{\pm 0.001}$)	0.379$_{\pm 0.001}$ (0.392$_{\pm 0.001}$)	0.379$_{\pm 0.001}$ (0.395$_{\pm 0.002}$)

from regularization. Interestingly, only batch negatives and adversarial loss are effective regularizers, and the model does not benefit from input noise and history dropout. This suggests that the dataset possibly includes enough types of user history, but lacks examples of corresponding target interests, because targets are harder for the model to overfit on with more negatives and adversarial loss. On MIND, no regularization is effective, and only slightly hurts the ranking performance. In-batch negatives are especially ineffective. This could be the result of the larger item catalogue, and the higher-quality impression logs with more accurate negative samples, meaning that SANNR can utilize more model capacity in user modeling to reduce model bias, and that extra sampled negatives only hurt performance because they are less accurate negatives.

Item projection ablation Table 12 shows performance of the SANNR variants that include the `Adapt (i)` component across embedding projection strategies. As also seen in Figure 9, on MIND, no projection strategy is effective, and we see that delta projection outperforms linear projection, while on NUMEND, linear projection is most effective. This pattern where increased projection capacity leads to worse performance on MIND suggests that the embedding projection layers are overfitting on MIND. And on NUMEND, the opposite happens where linear projection slightly outperforms the more heavy regularized delta projection. This is the exact opposite pattern from the regularization ablation study, where SANNR only benefits from regularization on NUMEND. This is likely caused by higher sparsity of user-item interactions in MIND,

Table 12: Ablation of projection in SANNR variants with self-attention. Performance is measured by val/test nDCG@10, subscripts show standard deviation over 3 repetitions, and bracketed scores show test performance.

	none	linear	delta projection
NUMEND			
3: SANNR (i+u)	$0.581_{\pm 0.000}$ ($0.573_{\pm 0.001}$)	$\mathbf{0.587_{\pm 0.000}}$ ($\mathbf{0.581_{\pm 0.001}}$)	$0.587_{\pm 0.001}$ ($0.580_{\pm 0.002}$)
4: SANNR (i+ID+u)	$0.593_{\pm 0.000}$ ($0.578_{\pm 0.001}$)	$0.596_{\pm 0.000}$ ($\mathbf{0.582_{\pm 0.000}}$)	$\mathbf{0.597_{\pm 0.001}}$ ($0.581_{\pm 0.000}$)
MIND			
3: SANNR (i+u)	$\mathbf{0.376_{\pm 0.002}}$ ($\mathbf{0.394_{\pm 0.003}}$)	$0.368_{\pm 0.000}$ ($0.389_{\pm 0.001}$)	$0.369_{\pm 0.002}$ ($0.388_{\pm 0.004}$)
4: SANNR (i+ID+u)	$\mathbf{0.382_{\pm 0.001}}$ ($\mathbf{0.393_{\pm 0.000}}$)	$0.371_{\pm 0.001}$ ($0.385_{\pm 0.001}$)	$0.373_{\pm 0.000}$ ($0.386_{\pm 0.000}$)

meaning that there are not enough examples per item to learn news embedding projections that generalize beyond the training distribution. In NUMEND, there are more dense user-item interactions, resulting in more stable embedding projections that generalize well to the test set. Interestingly, the regularization measures do not help regularize such embedding projections on MIND (the SANNR (i+$\cdots$) variants). This suggests that regularization mainly prevents overfitting of user modeling components, and does not help with overfitting of item projections, likely because the item sparsity dominates regularization in MIND. The fact that linear and delta projection perform nearly identically, shows that either delta projection still allows for too much capacity, or that linear projection is able to effectively preserve semantic space.

QKV projection ablation Table 13 shows ranking performance of all SANNR variants over the query, key and value projections used to obtain the query, key, and value matrices for self-attention. Differences in performance are very subtle, and volatile. Notably, QKV projection does not consistently outperform no projection, where semantic embeddings are directly used as attention matrices. Only for the SANNR (u) variant, we see that linear or delta QKV projection slightly outperforms no QKV projection, showing that QKV projection is *not* effective in SANNR when stacked with any other components that optimize parameters for recommendation, and only *slightly* effective when it is the only learnable component. Because self-attention is still very effective, as shown in Figure 9, this indicates that the main mechanism self-attention utilizes in frozen NNR is *embeddings augmentation* with *similar embeddings* as self-attention mostly relies on embedding similarity. Likely, this embedding augmentation resolves ambiguity in inferring underlying user interests through combining overlapping semantic topics. So, frozen NNR call for near parameter-

Table 13: Ablation of query, key, and value self-attention projection in SANNR. Performance is measured by val/test nDCG@10, subscripts show the standard deviation over 3 repetitions, and bracketed scores show test performance.

	none	linear	delta projection
NUMEND			
1: SANNR (u)	$0.580_{\pm 0.000}$ ($0.577_{\pm 0.000}$)	$0.582_{\pm 0.000}$ ($0.575_{\pm 0.001}$)	$\mathbf{0.584_{\pm 0.000}}$ ($\mathbf{0.578_{\pm 0.000}}$)
2: SANNR (ID+u)	$\mathbf{0.593_{\pm 0.000}}$ ($0.578_{\pm 0.001}$)	$0.591_{\pm 0.000}$ ($\mathbf{0.579_{\pm 0.000}}$)	$0.588_{\pm 0.001}$ ($0.575_{\pm 0.001}$)
3: SANNR (i+u)	$0.587_{\pm 0.001}$ ($0.580_{\pm 0.003}$)	$0.588_{\pm 0.001}$ ($\mathbf{0.582_{\pm 0.002}}$)	$\mathbf{0.589_{\pm 0.001}}$ ($0.580_{\pm 0.001}$)
4: SANNR (i+ID+u)	$\mathbf{0.596_{\pm 0.000}}$ ($\mathbf{0.583_{\pm 0.001}}$)	$0.591_{\pm 0.001}$ ($0.580_{\pm 0.002}$)	$0.591_{\pm 0.001}$ ($0.578_{\pm 0.001}$)
MIND			
1: SANNR (u)	$0.377_{\pm 0.001}$ ($0.393_{\pm 0.001}$)	$\mathbf{0.379_{\pm 0.000}}$ ($0.395_{\pm 0.000}$)	$0.378_{\pm 0.000}$ ($\mathbf{0.396_{\pm 0.001}}$)
2: SANNR (ID+u)	$\mathbf{0.382_{\pm 0.000}}$ ($0.393_{\pm 0.000}$)	$0.381_{\pm 0.000}$ ($0.394_{\pm 0.000}$)	$0.382_{\pm 0.000}$ ($\mathbf{0.395_{\pm 0.001}}$)
3: SANNR (i+u)	$\mathbf{0.369_{\pm 0.000}}$ ($\mathbf{0.388_{\pm 0.003}}$)	$0.366_{\pm 0.001}$ ($0.384_{\pm 0.003}$)	$0.369_{\pm 0.000}$ ($0.385_{\pm 0.001}$)
4: SANNR (i+ID+u)	$\mathbf{0.372_{\pm 0.001}}$ ($\mathbf{0.385_{\pm 0.001}}$)	$0.368_{\pm 0.001}$ ($0.385_{\pm 0.001}$)	$0.368_{\pm 0.001}$ ($0.383_{\pm 0.001}$)

free self-attention relying on semantic similarity.

5.2.4 Results: baseline comparison

Table 14 shows nDCG@10, precision@1 scores for the SANNR variants and baselines on MIND and NUMEND. Additionally, inference and training times for processing a thousand samples are listed for all recommenders apart from NaSE; training/inference compute for NaSE is equivalent to the LF baseline. Ranking and classification performance of all SANNR variants outperform all frozen NNR and LF baselines on NUMEND while maintaining cheap inference and training. While the unfrozen external NNR baselines would outperform all SANNR configurations, they come at the cost of increased compute of approximately a factor 100, *which makes them unsuitable for production systems*.

Frozen NNR outperform the LF baseline on NUMEND. This shows that user modeling components are capable of some level of task adaptation. On MIND, this is not the case with any poly attention user modeling strategies, and only with the NNR using self-attention. Among the frozen NNR, NRMS performs best, showing that multi-head self-attention is effective, even more so than learned user ID embeddings with single-headed self-attention. Frozen MINER

Table 14: Recommender performance measured by NDCG@10 and precision@1 of SANNR variants on NUMEND and MIND compared to frozen NNR, late fusion, and external* baselines from Iana et al. (2025) and Qi et al. (2022). The training and inference time column shows seconds to process 1000 examples. Recommender performance cells show validation performance, test performance in brackets, and standard deviation (from three repetitions) in subscript. The best scores per column and dataset split are boldfaced.

	Train | Infer	NUMEND N@10	NUMEND P@1	MIND N@10	MIND P@1
LF NaSE*	– | –	– –	– –	– ($0.390_{\pm 0.000}$)	– –
NAML*	$7.980s$ | $1.230s$	– –	– –	– ($0.419_{\pm 0.001}$)	– –
NRMS*	$11.390s$ | $2.160s$	– –	– –	– ($\mathbf{0.422_{\pm 0.003}}$)	– –
LSTUR*	$11.530s$ | $2.430s$	– –	– –	– ($0.422_{\pm 0.002}$)	– –
LF baseline	$0s$ | $0.009s$	$0.548_{\pm 0.000}$ ($0.545_{\pm 0.000}$)	$0.252_{\pm 0.000}$ ($0.262_{\pm 0.000}$)	$0.364_{\pm 0.000}$ ($0.386_{\pm 0.000}$)	$0.175_{\pm 0.000}$ ($0.178_{\pm 0.000}$)
Frozen NAML	$0.043s$ | $0.009s$	$0.559_{\pm 0.001}$ ($0.547_{\pm 0.000}$)	$0.302_{\pm 0.001}$ ($0.290_{\pm 0.000}$)	$0.363_{\pm 0.000}$ ($0.384_{\pm 0.000}$)	$0.174_{\pm 0.000}$ ($0.174_{\pm 0.001}$)
Frozen MINER	$0.043s$ | $0.009s$	$0.560_{\pm 0.000}$ ($0.546_{\pm 0.000}$)	$0.303_{\pm 0.001}$ ($0.289_{\pm 0.001}$)	$0.363_{\pm 0.000}$ ($0.385_{\pm 0.000}$)	$0.174_{\pm 0.000}$ ($0.175_{\pm 0.001}$)
Frozen NRMS	$0.063s$ | $0.022s$	$0.582_{\pm 0.000}$ ($0.573_{\pm 0.001}$)	$0.325_{\pm 0.000}$ ($0.313_{\pm 0.002}$)	$0.381_{\pm 0.001}$ ($0.392_{\pm 0.002}$)	$0.186_{\pm 0.001}$ ($0.176_{\pm 0.001}$)
Frozen LSTUR	$0.087s$ | $0.022s$	$0.580_{\pm 0.000}$ ($0.567_{\pm 0.001}$)	$0.324_{\pm 0.001}$ ($0.308_{\pm 0.002}$)	$0.371_{\pm 0.001}$ ($0.372_{\pm 0.002}$)	$0.178_{\pm 0.002}$ ($0.157_{\pm 0.001}$)
1: SANNR (u)	$0.063s$ | $0.023s$	$0.585_{\pm 0.000}$ ($0.578_{\pm 0.001}$)	$0.329_{\pm 0.000}$ ($0.320_{\pm 0.001}$)	$0.378_{\pm 0.001}$ ($0.395_{\pm 0.000}$)	$0.184_{\pm 0.001}$ ($\mathbf{0.186_{\pm 0.001}}$)
2: SANNR (ID+u)	$0.077s$ | $0.019s$	$0.593_{\pm 0.000}$ ($0.578_{\pm 0.000}$)	$\mathbf{0.342_{\pm 0.000}}$ ($\mathbf{0.326_{\pm 0.000}}$)	$\mathbf{0.382_{\pm 0.001}}$ ($0.393_{\pm 0.000}$)	$\mathbf{0.190_{\pm 0.001}}$ ($0.186_{\pm 0.000}$)
3: SANNR (i+u)	$0.079s$ | $0.024s$	$0.589_{\pm 0.001}$ ($0.580_{\pm 0.001}$)	$0.331_{\pm 0.000}$ ($0.322_{\pm 0.001}$)	$0.366_{\pm 0.003}$ ($0.386_{\pm 0.003}$)	$0.170_{\pm 0.001}$ ($0.176_{\pm 0.002}$)
4: SANNR (i+ID+u)	$0.104s$ | $0.025s$	$\mathbf{0.596_{\pm 0.001}}$ ($\mathbf{0.582_{\pm 0.001}}$)	$0.339_{\pm 0.002}$ ($0.323_{\pm 0.001}$)	$0.372_{\pm 0.000}$ ($0.387_{\pm 0.001}$)	$0.174_{\pm 0.001}$ ($0.176_{\pm 0.001}$)

outperforming NAML shows that poly attention is effective, confirming that the 'multiple aspects of user interests' hypothesis from MINER holds for frozen item encoders on NUMEND. On MIND, this is clearly not the case, meaning that the frozen item encoder limits the expressiveness of news encodings to the extent that no multiple aspects of user interests can be highlighted by poly attention.

On NUMEND, all SANNR variants outperform all frozen NNR baselines, while maintaining cheap inference and training. Specifically, SANNR achieves up to a 0.009 test nDCG@10 point gain, and up to a 0.013 test precision@1 point gain compared to frozen NRMS on NUMEND. On MIND, not all SANNR variants outperform all frozen NNRs. Notably, frozen NRMS performs well. Looking at test performance, the best SANNR variant (which is SANNR (u)) performs 0.003 test nDCG@10 points better and 0.01 test precision@1 better. In combination with ablation study results, this shows that frozen NNRs in all cases benefit from thoughtful architectural selection: a suitable Frankenstein of NNR components. Furthermore, for NUMEND, we see that frozen NNRs also benefit from additional embedding projection and regularization that SANNR implements.

Among the SANNR variant performance on NUMEND, the best ranker is SANNR (i+ID+u), with a 0.002 test nDCG@10 point gain over the next best. The best top 1 classifier is SANNR (ID+u), which performs best by a larger margin of a 0.003 precision@1 point gain. Both best performing variants leverage user ID embeddings, which shows that user preference embeddings add valuable capacity for task adaptation. On MIND, we see that those user ID embedding variants get high validation performance, but low test performance, indicating an increase in trend dependence of these variants. So, SANNR benefits from embedding projection, but only for ranking performance and only on NUMEND. Likely, news embedding projection creates a better ranker through restructuring of semantic space, allowing for cross-topic relevance scoring.

Lastly, we see that SANNR outperforms the LF NaSE baseline by as much as 0.005 test nDCG@10 on MIND while requiring equivalent inference compute. Possibly, this performance gain would be even greater on NUMEND because SANNR projection and regularization are more effective. This result highlights a trade-off between task adaptation through fine-tuning of sentence encoders (NaSE), and task adaptation through learnable NNR components. Fine-tuning SE is expensive, but generalizes to arbitrary future examples, while training NNR on top of frozen sentence encoders is cheap and makes for a better ranker, but arguably requires more frequent re-training to maintain performance across time, especially the variants using user ID embeddings.

5.3 Task adaptation characteristics

Considering user behavior, popularity bias, news categories, user history length, and generalization to future dates on nu.nl; in what ways does SANNR's task adaptation change recommendation behavior compared to the frozen semantic baseline?

5.3.1 Experimental setup

We further analyze how SANNR achieves task adaptation by considering various aspects of the recommender task: (i) collaborative user behavior, (ii) popularity bias, (iii) category-dependence, (iv) user history length, and (v) recommender generalization across time. We analyze collaborative user behavior at category-level using co-occurrence probability measures, popularity bias using average popularity in top-k recommendations, category-dependence by looking at recommender performance per category, user history length by plotting recommender performance over corresponding history length, and generalization across time by plotting performance over days since SANNR was last trained. We list the details of each experiment in the same order. For all analyses, we compare SANNR variants against the semantic baseline because this baseline implements no task adaptation, meaning that this allows us to best answer how SANNR optimizes semantic embeddings for recommendation. We only include results on the NUMEND dataset because we require full log data for this analysis.

Behavioral alignment To analyze behavioral alignment of SANNR, we compare similarity rankings of *predicted similarity* between article categories with *measured co-occurrence probability* of articles like we did in Subsection 3.7. Additionally, to shed light on document frequency and length bias within clusters that can play a role in probability measures as well as semantic similarity, we compare all similarity rankings against rankings from user and article counts.

Popularity bias We measure popularity bias by comparing distributions of item popularity, which we represent as the total proportion of clicks within the last 6 hours. Specifically, we compare the popularity distribution in candidate pools, the top 5 semantic baseline recommendations, and the top 5 recommendations of all SANNR variants. We plot each distribution with a box plot, and show the precise values in a table that lists the 5th, 25th, median, 75th, and 95th popularity percentiles. This experiment sheds light on task adaptation by learning popularity.

Category dependence This experiment compares recommender ranking performance per category to learn about the news categories for which SANNR does task adaptation. We report average nDCG@10 testing ranking performance (including standard deviation) for the semantic baseline and 4 SANNR variants. We aggregate this per-category performance on all impression logs in the NUMEND test split by grouping by the categories of the clicked articles. This means that the average ranking score, and corresponding standard deviation, is aggregated from a single optimization and inference run, and not multiple repetitions. Together with ranking scores per news category, we report scores per subcategory to better shed light on specific types of articles that allow for effective task adaptation.

User history length We measure performance over user history length by again aggregating ranking performance of individual examples in the test split of NUMEND over history length. We report a regular line plot of mean nDCG@10 ranking performance obtained from bins of users. Each bin contains approximately the same amount of impression log targets, and groups users with similar user history lengths. So, the bin with the longest median user history length contains a wider range of user history lengths because longer history lengths are more rare. This type of binning reduces noisy estimates of average nDCG@10 for longer history lengths caused by fewer examples. Complementing this line plot, we show nDCG@10 score distributions for the 20th, 50th, and 80th percentile active users. Users in the 80th percentile for example have a history length that is longer than 80% of users. This extends on previous data analysis.

Generalization across time Lastly, we measure recommender performance over days since trained. So, here we aggregate impression log targets by their publication date. We treat the first day in the testing split as 0 days since trained, even though the in-between three-days-long validation set means that it's actually been 3 days. We do so because the model could theoretically have overfit on the validation split through hyper parameter tuning. We design this experiment to find how frequent SANNR needs to be re-trained to maintain performance. We again show a line plot of average nDCG@10 values.

5.3.2 Results: behavioral alignment

Table 15 shows Spearman's rank correlation between *assigned* ranking of category similarity by the content of their news articles, and *measured* ranking of category co-occurrence in user interaction sequences. Additionally, this table includes the correlation to user and article counts used as similarity ranking. For example, categories with the highest article counts are ranked first using this method. High correlations imply that the ranked lists are similar.

The trained adapter rankings have a higher correlation with bi-directional co-dependence measurement nPMI compared to embedding similarity, indicating that these methods better model behavioral item similarity. The item adapter models this behavioral item similarity slightly better than user adapters, indicating that item-level adaptation is more effective at modeling behavioral item similarity than user-level adaptation. However, the difference is small, showcasing that asymmetrical user-level similarity ranking can actually capture most of the item-level behavioral similarity.

From directional PMI and attributed conditional probability, it is clear that semantic similarity is on par with or has better correlation than trained adapters in modeling directional similarity. One explanation for this is the training task of the trained adapters, which considers chronological ordering, and predicts future interactions based on past interactions, as opposed to the conditional probabilistic methods modeling cross-session non-chronological conditional occurrence in user interactions. Furthermore, attributed directional probability

Table 15: Spearman rank correlations between measured and recommender rankings of news topic similarity in the NUMEND test set. Recommenders rank topic (directional) similarity between topics A and B by predicting relevance of topic B articles given a reading history of all articles of topic A. Measured topic similarity includes (n)PMI and directional PMI, that each measure probability of topics co-occurring in user interaction sequences, acting as a proxy for behavioral similarity. Log counts are also included, that create ranked lists by counts of users that interacted with a topic or counts of articles in a topic. SANNR correlates more strongly with co-occurrence-based measures, suggesting it better captures cross-topic behavioral similarity, whereas the baseline's strong alignment with count rankings points to a document-length bias.

	Ranked by	Measurements		Counts	
		NPMI	$PMI_{A \to B}$	# users	# articles
Recommender	Baseline	0.29	0.25	0.46	0.47
	SANNR (i+u)	0.58	0.32	0.10	-0.00
	SANNR (u)	0.53	0.31	0.09	0.03
Counts	# users	0.07	0.11	1.00	0.87
	# articles	-0.23	0.03	0.87	1.00

has a comparatively high correlation to user and article counts. So, this is likely part of the reason why embedding similarity has the highest correlation with this ranking metric. This leaves the asymmetrical PMI measurement, where cosine similarity is on-par with the trained adapters.

Compared to other ranking methods, plain embedding similarity ranking has a strong correlation to the user and articles count ranking. This means that adding more embeddings to a cluster boosts similarity scores to either other clusters with many embeddings or all embeddings. In other words, this is a document length bias. In behavior space, this means that a larger embedding clusters or longer articles get higher relevance scores to longer reading histories. By contrast, the fine-tuned encoders have almost no correlation with user and article counts, hinting that this correlation is undesired in modeling behavior space.

5.3.3 Results: popularity bias

Figure 10 shows the distribution of popularity of the items in the candidate pool, and the top 5 recommendations by all the SANNR variants and the semantic baseline. Table 16 lists the numeric results from the figure. It is clear that there are no large differences in the average or median popularity in the top-5 recommendations of any algorithm and the candidate pool. Compared to the parameter-free baseline, SANNR recommends slightly less popular items (in the top 5). This suggests that negative sampling from the popularity distri-

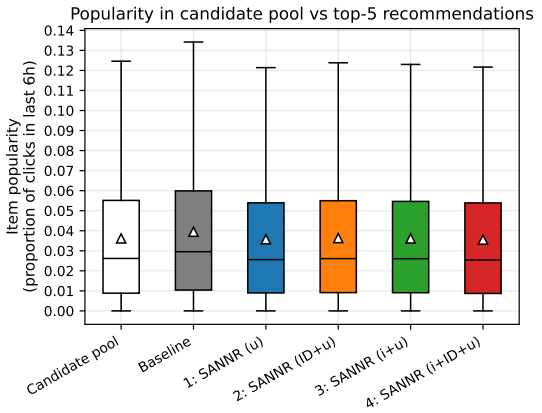


Figure 10: Distribution of recommended-item popularity for each SANNR variant versus the candidate pool. Boxes show [P25,P75], whiskers show [P5,P95], lines show the median, and the triangles show the mean of each variant. Colors map to the same algorithm as the other figures.

Table 16: Popularity distribution of the top-5 recommendations per variant compared to the candidate pool from Figure 10. Highest article popularity scores are boldfaced, which shows which models incorporate popularity the most.

	mean	median	P5	P25 (Q1)	P75 (Q3)	P95
Candidate pool	0.0362	0.0262	0.0008	0.0088	0.0551	0.1056
Baseline	**0.0395**	**0.0295**	**0.0009**	**0.0104**	**0.0599**	**0.1144**
1: SANNR (u)	0.0357	0.0256	0.0008	0.0090	0.0539	0.1060
2: SANNR (ID+u)	0.0363	0.0261	0.0008	0.0091	0.0550	0.1072
3: SANNR (i+u)	0.0361	0.0260	0.0008	0.0091	0.0546	0.1067
4: SANNR (i+ID+u)	0.0356	0.0254	0.0008	0.0087	0.0539	0.1054

bution and in-batch negatives prevented SANNR from learning to recommend popular items as opposed to learning user preferences from user histories and ID's. Moreover, results show that the parameter-free baseline top 5 recommendations contain more popular items than the popularity sampled candidate pool *and* the top 5 SANNR recommendations. This might suggest that the baseline recommender is more biased in reinforcing popularity bias present in user histories than any SANNR variant.

5.3.4 Results: category dependence

Table 17 shows recommendation ranking performance (nDCG@10) across category labels, with corresponding standard deviation. Figure 11 shows more in-detail performance over subcategories. All SANNR variants outperform the baseline on almost all categories and subcategories. The category SANNR im-

Table 17: Mean (± std) nDCG@10 per parent category. Best per column in bold.

	Algemeen	Sport	Media en Cultuur	Economie	Overig
Baseline	$0.573_{\pm 0.2596}$	$0.587_{\pm 0.2619}$	$0.536_{\pm 0.2521}$	$0.470_{\pm 0.2377}$	$0.498_{\pm 0.2581}$
1: SANNR (u)	$0.567_{\pm 0.2530}$	$\mathbf{0.636_{\pm 0.2591}}$	$0.557_{\pm 0.2608}$	$0.543_{\pm 0.2517}$	$0.558_{\pm 0.2575}$
2: SANNR (ID+u)	$0.570_{\pm 0.2553}$	$0.633_{\pm 0.2585}$	$0.576_{\pm 0.2619}$	$0.524_{\pm 0.2500}$	$0.567_{\pm 0.2631}$
3: SANNR (i+u)	$\mathbf{0.588_{\pm 0.2524}}$	$0.628_{\pm 0.2617}$	$0.553_{\pm 0.2624}$	$0.537_{\pm 0.2494}$	$0.572_{\pm 0.2644}$
4: SANNR (i+ID+u)	$0.571_{\pm 0.2519}$	$0.627_{\pm 0.2594}$	$\mathbf{0.576_{\pm 0.2626}}$	$\mathbf{0.547_{\pm 0.2521}}$	$\mathbf{0.591_{\pm 0.2660}}$

proves the least on is 'general': 0.015 point increase at best, but even a point decrease for the majority of SANNR variants. For the rest of the categories, point increases range from 0.040 (media and culture) to 0.093 (other).

So, the 'general' news category stands out the most: only SANNR (i+u) outperforms the semantic baseline. This news category is the most generic one, that also contains the most articles that were published on the front page. From looking at the subcategories, it is clear that SANNR is not well-optimized for domestics, politics and crime, which are large clusters within general news. We think this shows that SANNR needs more frequent re-training for maintained accuracy on such trend-driven topics. SANNR (i+u) compensates for this with increased ranking performance of articles about American politics, showing that this category benefits from projected embedding space or learned fusion that only SANNR (i+u) implements.

The 'Media and culture' and 'other' news categories also stand out, but for a different reason. All SANNR variants outperform the baseline for these categories, with item projection variants performing better on 'other', and user ID embedding variants on 'media and culture'. Specifically, we see that embedding projection works well for subcategories such as 'health' and 'remarkable', while user ID embeddings slightly help with ranking 'gossips', 'media', or 'movies' articles.

5.3.5 Results: user history length

Figure 12 shows average recommender ranking performance over user history length (Subfigure 12a), and performance across three groups of users by their activity percentile (Subfigure 12b).

From Subfigure 12a, it is clear that recommender performance first increases when user history length increases. However, for user history lengths of more than ten articles, recommender performance decreases for increasing history length. This holds true for all SANNR variants and the baseline recommender. We think that this shows that shorter histories are often more carefully selected according to niche interests, making them more indicative of interests, and that active users read less relevant articles just because there were no more highly relevant articles available in the catalog, also leading to longer reading histories that are less indicative of interests.

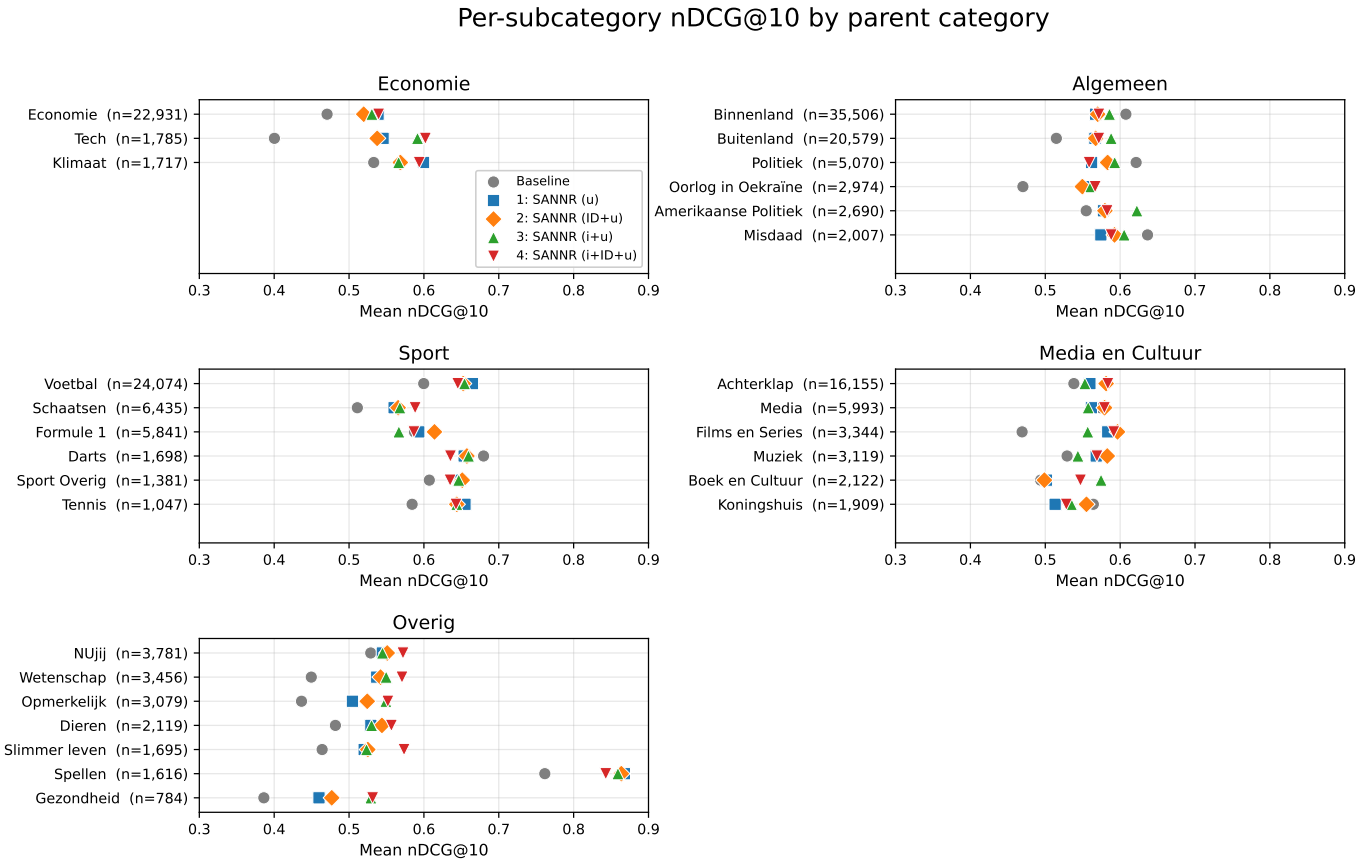
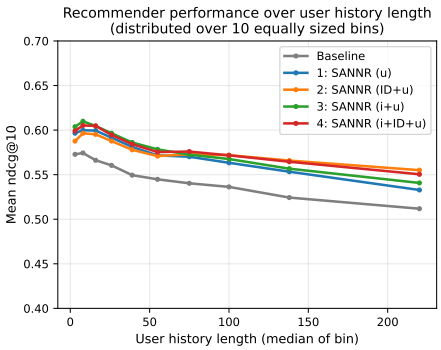
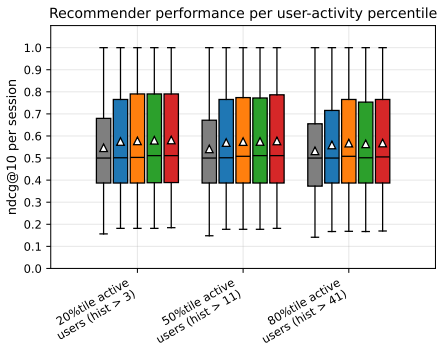


Figure 11: Per-subcategory nDCG@10 grouped by parent category. Each panel shows the subcategories within one parent; markers distinguish variants and the y-axis shows the per-subcategory sample size.

From Subfigure 12b we see consistent recommender ranking performance across user activity group. When more closely examining the difference, we see that the ranking performance distributions do slightly change for all five recommenders: it becomes less likely for recommenders to achieve the highest nDCG for more active user groups. The medians and lower bounds stay more stable. This indicates that niche active users have less indicative user histories for their interests because they deplete the article pool more. Lastly, we see that the top parts of the baseline score distributions are more narrow. We think this indicates that SANNR optimized for examples that the baseline likely already scored above median ranking performance on.



(a) Mean nDCG@10 per user-history-length bin. 10 bins each contain a similar amount of sessions, sliced from history length sorted user list.



(b) Session-level nDCG@10 distribution per user-activity percentile (more active than 20%, 50%, 80% of users). Box shows [P25,P75], whiskers show [P5,P95], line shows the median, and the triangle shows the mean. Colors map to the same algorithm as in the left figure.

Figure 12: Recommender ranking performance by user history length on NUMEND. Subfigure 12a shows the trend across the full range. Subfigure 12b shows performance distributions for three user subgroups.

5.3.6 Results: generalization across time

Figure 13 shows recommender test performance on NUMEND over the days elapsed since the models were trained, which is the amount of days passed since the last example in the validation set. Recommender performance fluctuation across days elapsed in the figure is consistent across algorithms, even the parameter-free baseline. This likely suggests that for some days the recommendation task was more difficult, and that SANNR variants do not need to be frequently re-trained to achieve performance that is better than the baseline. However, these results do not include the validation set. Earlier results suggested that SANNR does achieve much better performance on the three-day-long validation set, after which performance degraded. So, we suspect that SANNR does capture short-term (around 3 day long) trends, but also more stable user interests which is indicated by the stable relative-to-baseline performance in Figure 13.

5.4 User study

Do users perceive personalized recommendations by SANNR as more relevant and interpretable compared to the semantic recommender?

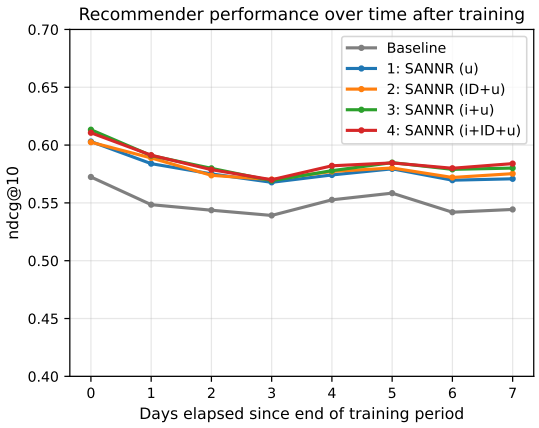
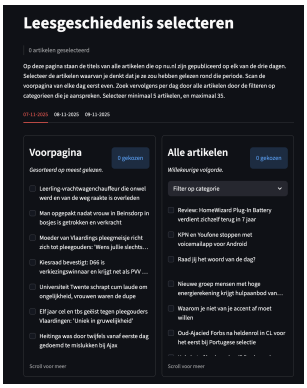
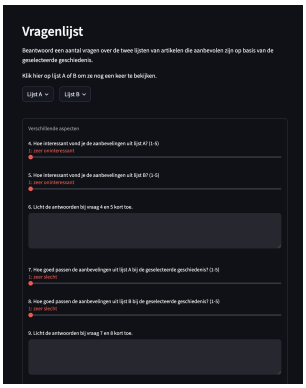


Figure 13: nDCG@10 per day after the end of the validation dataset. All algorithms follow the same trend in recommender performance change over days, even the trained SANNR variants.

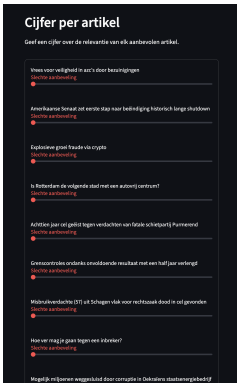
5.4.1 Experimental setup



(a) History selection



(b) List-wise questionnaire



(c) Per-article questionnaire

Figure 14: The three user study phases. In Phase 14a, users select 5–30 articles they would read. In Phase 14b, users rate two blind, counterbalanced top-10 lists (baseline and SANNR) on relevance, interpretability, diversity, and consistency. In Phase 14c, users label relevance of each article from the union of both lists on a 1–5 Likert scale.

To evaluate user-perceived recommendation relevance, we conduct a within-subjects user study ($N = 12$) comparing the best SANNR variant without news modeling and user ID embeddings against the baseline recommender. Click logs are noisy (accidental clicks), biased (non-homepage articles are infrequently

clicked), and often do not represent underlying user interests clearly. The user study allows for more unbiased evaluation of recommendation relevance by asking users to assign a relevance label to each recommended item, regardless of their position. Noise is mitigated because users adjust relevance labels until they are satisfied with them. Lastly, intent of users is better uncovered with open-ended questions in the questionnaire, and by interviewing users about their thought process during the user study.

The user study is formatted as a survey that consists of three phases: reading history selection (illustrated in Subfigure 14a), recommendation list-wise questionnaire (illustrated in Subfigure 14b), per-article relevance assessment (illustrated in Subfigure 14c). Reading history is selected from articles published in the last three days of the training split, and the recommendation candidate pool is the first day of the test split. Temporal splits from NUMEND's train/test splits allow for comparison with offline results on NUMEND. One caveat with this approach is that user interests have shifted, meaning that users can no longer reliably assess historical relevance. For instance, sports readers may historically have been likely to also read politics articles because politics was trending during the NUMEND sampling period. Such trend-driven relevance is hard to judge in hindsight because the trend likely has faded. This user study therefore implicitly focuses more on stable, long-term, interests. The survey is custom-written using the Streamlit library for rendering the web interface with Python. Recommender inference is done locally (with sub-second speed) on a Macbook Air M3 CPU. We continue this section with describing the three user study phases, and end with defining the metrics and other results that we report.

Phase 1: history selection The first user study phase is the reading history selection (Subfigure 14a). Here we ask users to select 5 to 30 articles, based on the criteria if they think they would read this article. We make clear to users to be strict on selecting articles to reduce the selected reading history lengths to representative amounts to the NUMEND median, and to ensure the user histories are representative of their interests. Articles are grouped per day, and within each group, users first see the articles that were published on the home page to mimic article exposure on nu.nl, and then see all articles (about 100 articles), where they can filter based on topic to more quickly find articles that they like. The 'all articles' section is shuffled to prevent alphabetic or temporal positional bias. Users can only proceed with 5 or more selected articles.

Phase 2: list-wise recommendation questionnaire The second user study phase is the list-wise questionnaire (Subfigure 14b). Here we generate two lists of top 10 recommendations from the candidate pool based on the selected reading history in the previous step. One list is created with the baseline, and the other with SANNR. Users do not know which list is created by which recommender. The candidate pool consists of articles with publication dates in the first day of the *testing* split of NUMEND. The order of the two lists alternates

per user to account for position bias. About the lists (list A and list B), users are asked:

Perceived relevance.

1. How interesting did you find the recommendations in list A?
2. How interesting did you find the recommendations in list B?
3. Briefly explain your answers to Q1 and Q2.

Interpretability (history match).

4. How well do the recommendations in list A match the selected history?
5. How well do the recommendations in list B match the selected history?
6. Briefly explain your answers to Q4 and Q5.

Diversity.

7. The recommendations in list A are diverse.
8. The recommendations in list B are diverse.

Consistency (presence of clearly inappropriate items).

9. List A contains articles that don't appeal to me at all.
10. List B contains articles that don't appeal to me at all.

The first three questions measure overall perceived relevance, and are designed to be especially subjective to collect open-ended user feedback about the recommendation lists. The next three questions measure perceived interpretability, which is a more focused way of measuring perceived relevance of recommendations to user histories, and uncovers perceived interpretability, which is an important pillar within the DPG Media organisation. The next two questions measure perceived diversity in the recommendation lists, which we included because this potentially highlights the side-effect of algorithms that are more exploitative in leveraging user histories for recommendations. Finally, the last two questions evaluate if users think that some recommendations are especially irrelevant, allowing us to further explain user-labeled perception of relevance. For example, users could be very put-off by few bad recommendations, causing them to find the entire list less relevant, regardless of the other recommendations.

Phase 3: per-recommendation relevance assessment We take the union of the two top 10 article recommendations, shuffle them, and ask participants to label if they would read each article (on a Likert scale from 1 to 5), with the additional instruction that a score of higher than 3 means that they see themselves potentially reading it at some point during a day. Subfigure 14c shows the screenshot of this step. The goal of this evaluation is to find detailed differences between the two recommendation lists, allowing for evaluation using

derived metrics such as precision and nDCG@10. nDCG@10 is computed from the Likert ratings, and precision from binarized (at > 3). This phase serves two purposes. First, it allows us to compare user-perceived relevance of lists with exact amounts of perceived article relevance per list. Second, we expect the per-article derived metrics to be more accurate measurements of perceived list relevance than list-wise Likert ratings because the per-article relevance labels avoid cognitive difficulty of evaluating entire lists.

Analysis In summary, all metrics that we measure in the user study are:

Per-article rating metrics (measured on the page illustrated in Subfigure 14c)

1. Precision@10 $[0, 1] \uparrow$
2. nDCG@10 $[0, 1] \uparrow$

List rating metrics (measured on the page illustrated in Subfigure 14b)

3. Interest $[1, 5] \uparrow$
4. Interpretability $[1, 5] \uparrow$
5. Diversity $[1, 5] \uparrow$
6. Dislike $[1, 5] \downarrow$

Per metric and algorithm, we show a boxplot of the score distribution over users, where we also include a visual indication of pairwise within-user scores. We test for paired differences between the algorithms using the Wilcoxon signed rank test because this method doesn't require the normal distribution assumption, and therefore allows for proving statistical relevance using fewer users. We scale the list-wise Likert scores in the boxplot to $[0, 1]$, so they share the same axis with the precision and nDCG metrics. Complementary to the boxplots, we list detailed metric and metric ranges in a table. Lastly, we report cherry-picked per-user scores with their open-ended answers to give an indication of strengths and weaknesses of both algorithms, and remaining challenges.

5.4.2 Summary results

Figure 15 shows the summary results of the user study. This includes the boxplots and an indication of pairwise within-user scores for two metrics that are derived from the per-article Likert relevance assessments, and four that are derived from the list-wise questionnaire. Table 18 lists the results in the figure in more detail. Furthermore, statistical significance is listed in both the figure and the table. The results show a precision improvement of SANNR recommendations compared to the baseline algorithm recommendations of $0.45 \rightarrow 0.60$ with a statistical significance of $p < 0.05$. nDCG and dislike rating also suggest that SANNR recommendations are more relevant, but only for $p < 0.1$. The remaining interest ($p = 0.796$) and interpretability ($p = 0.590$) rating metrics

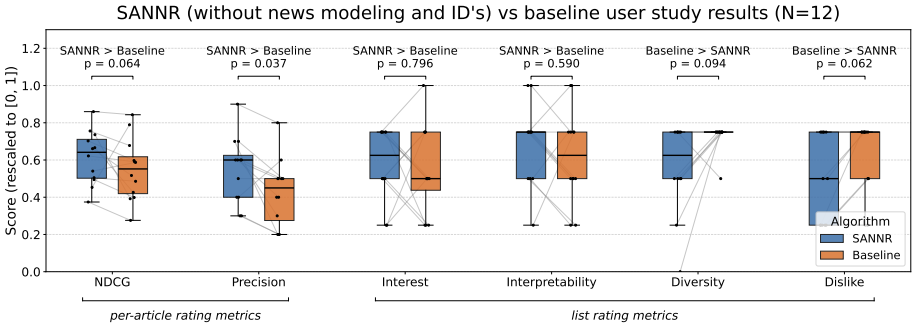


Figure 15: Box plots of scaled scores ($[0-1]$) comparing labeled recommendation relevance in a user study ($N = 12$) between lists generated by SANNR and the baseline. nDCG and precision are derived from the per-article relevance assessment ($1-5$ labels); the list rating metrics are labeled ($1-5$) at list granularity. Each metric includes a Wilcoxon signed-rank test measuring statistical significance of the system difference. Overall, SANNR recommendations are rated as more relevant than the baseline, but less diverse.

are statistically insignificant, but have medians that favor SANNR recommendations.

Both per-article metric results indicate that SANNR scores higher than the baseline algorithm: about 30% on precision and about 15% on nDCG in relative terms. Specifically, the median precision of 0.6 of SANNR indicates that *more than half* of SANNR recommendations are rated relevant across most users, while this is not the case for the baseline recommender with a median precision of 0.45. Among these two metrics, only the precision metric is statistically significant. Likely, this is the result of nDCG taking into account the full range of labels $[1, 5]$, making it more specific than the binary precision metric, which reduces tie scores and therefore reduces pairwise statistical significance. Table 18 shows this increase of ties: precision has two while nDCG has none.

The list-wise 'interest' assessment ($p = 0.796$) shows no statistically significant improvement of SANNR over the baseline, despite the per-article precision win. Notably, the contrasting list-wise 'dislike' metric reaches $p = 0.062$, indicating that users dislike the worst SANNR recommendation less than the worst baseline recommendation for $p < 0.1$. This pattern likely suggests that participants experience cognitive difficulty with labeling list-wise relevance where users have to compare the relevance and order of at most 20 articles. The dislike rating for instance is a comparatively simpler judgment for participants because they only need to locate and label one article per list. Following the same observations, the interpretability metric ($p = 0.590$) likely shows no statistically significant result for the same reason, although we have no simpler metric to directly compare it to.

Table 18: Per-user paired comparison of SANNR versus the baseline, corresponding to Figure 15. Scores are summarized as median [Q1, Q3]; W/T/L counts users for whom SANNR scored higher than, tied with, or lower than the baseline; p values come from a two-sided Wilcoxon signed-rank test. Bold indicates $p < 0.05$, with no correction for multiple comparisons.

Metric	SANNR	Baseline	W/T/L	p
NDCG	0.64 [0.50, 0.71]	0.55 [0.42, 0.62]	10/0/2	0.064
Precision	0.60 [0.40, 0.62]	0.45 [0.28, 0.50]	9/2/1	**0.037**
Interest	3.5 [3.0, 4.0]	3.0 [2.8, 4.0]	7/0/5	0.796
Interpretability	4.0 [3.0, 4.0]	3.5 [3.0, 4.0]	7/2/3	0.590
Diversity	3.5 [3.0, 4.0]	4.0 [4.0, 4.0]	1/5/6	0.094
Dislike	3.0 [2.0, 4.0]	4.0 [3.0, 4.0]	0/7/5	0.062

Although the diversity metric is also a list-wise score, it favors the baseline recommendations with a median user rating of 4 compared to 3.5 with a relatively large p of 0.094. This suggests that SANNR is more exploitative in inferring interests from user histories. The stronger signal here ($p = 0.094$ vs $p = 0.796$), likely suggests that the task is less difficult for users to evaluate, and/or that participants observe a clear and consistent difference in diversity. Notably, the two score distributions differ: almost all users assigned a rating of 4 for baseline recommendation diversity, while users assigned a larger range of scores for SANNR recommendation diversity, usually between 2 and 4. This could suggest that SANNR recommendation diversity is more dependent on selected reading history of users than the baseline recommender since this is the only factor that changes across users.

So, for the average participant, the recommendations generated by SANNR outperformed those by the baseline algorithm on relevance, confirming that SANNR models user interests that generalize to future user clicks. This user study also shows that the algorithm functions with reading histories selected with different exposure bias. Furthermore, the relevance assessments in this user study are influenced by different (current-day) trends than present the training and testing distribution (historic trends), indicating that SANNR likely models stable trend invariant interests.

5.4.3 Per-user results

Table 19 shows per-user user study results and summaries of their feedback. Firstly, we see that most users feel that SANNR ranks the lists better. Second, we see counterintuitive user feedback: for example, one user finds the semantic ranked list more relevant because it plainly contains more relevant articles, while precision does not reflect this. Third, we see that SANNR recommendations are often found more relatable to selected reading history. Fourth, we see that many users find the baseline recommendations more interesting because they

Table 19: Per-user results of the user study (N=12) on nu.nl articles published in November (reflecting the SANNR test set). Scores show SANNR | semantic baseline. For each user, we summarized their feedback. The first column shows which recommender was used for the first list. nDCG@10 and precision@10 are derived from per-article relevance labels, and the rest are Likert ratings.

First	nDCG@10	Precision@10	Interest	Diversity	Interpretability
Semantic	0.540 | 0.587	0.5 | 0.5	3 | 4	4 | 4	5 | 5
	Feedback summary: baseline was more diverse, and more relevant. Both reflect their history.				
SANNR	0.756 | 0.678	0.6 | 0.5	4 | 3	4 | 3	4 | 3
	Feedback summary: Baseline had less irrelevant regional news. Baseline reflects personal interests, and SANNR *reflects news interests.*				
Semantic	0.622 | 0.590	0.6 | 0.5	2 | 4	3 | 4	2 | 4
	Feedback summary: Baseline contains more relevant articles. Baseline better reflects specific American politics interests.				
SANNR	0.374 | 0.276	0.3 | 0.2	4 | 3	3 | 4	4 | 3
	Feedback summary: Both equally relevant, but SANNR *was ranked better, with a more relevant top recommendation.*				
Semantic	0.505 | 0.486	0.4 | 0.4	4 | 3	4 | 4	4 | 3
	Feedback summary: Both equally relevant, but SANNR *was ranked better to reflect their interests and history.*				
Semantic	0.666 | 0.426	0.6 | 0.4	2 | 3	2 | 4	4 | 3
	Feedback summary: Baseline has 3 relevant items and SANNR *2. They could not link many articles from the baseline to the history.*				
SANNR	0.661 | 0.517	0.6 | 0.3	3 | 2	4 | 4	4 | 4
	Feedback summary: Both lists felt monotonous; they wanted more of a hook to start reading. They could not recall their reading history well, but the recommendations seem somewhat relevant.				
Semantic	0.860 | 0.843	0.9 | 0.8	4 | 5	4 | 4	4 | 5
	Feedback summary: Both lists were interesting, but the baseline's ranking is better. Each list had some nice articles the other lacked.				
SANNR	0.453 | 0.789	0.4 | 0.6	3 | 4	1 | 4	3 | 4
	Feedback summary: SANNR *showed a lot of football, not all of interest, while the baseline was diverse. Both reflected their history well, but the baseline reflected more diverse articles.*				
Semantic	0.736 | 0.392	0.7 | 0.2	4 | 2	3 | 4	3 | 2
	Feedback summary: Baseline recommended odd articles (skating, very American defense news) despite their tech/science history. SANNR *was more interesting with no sports.*				
SANNR	0.495 | 0.399	0.3 | 0.2	3 | 2	3 | 4	3 | 2
	Feedback summary: SANNR *had some relevant politics and some unappealing topics. The baseline appealed even less. Only about 20–40% of their interests (politics, some tech/science/music) seemed reflected.*				
SANNR	0.703 | 0.597	0.7 | 0.5	4 | 3	4 | 4	5 | 4
	Feedback summary: SANNR *surfaced more topics that appeal to the user (e.g. Sinterklaas). Baseline recommended ice skating while they selected no sports.*				

are more diverse, even though some are not relatable to reading history. Lastly, it is clear that there is a position bias at play: people overall find the first list more relevant.

6 Discussion

This work explored the feasibility of keeping news encoders in two-tower NNR frozen in order to save compute compared to end-to-end training of news encoders with NNR, and to outperform semantic news encoders that are not task adapted. To this end, we proposed Semantically Adapting Neural News Recommender (SANNR) that implements light-weight cache-preserving task adaptation of semantic embeddings through carefully selected regularized NNR user modeling components and embedding projection components. SANNR is the first NNR to consider the unique design constraints that come with keeping news encoders frozen.

6.1 Main findings

Semantic baseline Our findings on sentence encoder results imply that the amount of required task adaptation differs per dataset. MIND shows to be more in-distribution for the largest multilingual sentence encoders than NUMEND, which we derived from the result that NUMEND benefits from the smaller news encoders that implement more multilingual adaptation steps than just pre-training on a mixed-language dataset. Because these lower capacity models outperform high capacity models, we argue that there is further room for task adaptation on news texts on NUMEND than on MIND. We argue this could explain why embedding projection of SANNR is more effective on NUMEND, together with the difference in item sparsity.

Architecture for frozen news encoders Frozen NNR benefit from different components than the end-to-end trained NNR from the literature, and the best set of components depends on the dataset. Our ablation study on SANNR showed that self-attention and ID embeddings are the most effective on both datasets, but that history fusion using poly or additive attention (used in many NNR such as MINER, NRMS, NAML) performs poorly, likely because this mechanism potentially factors away non-trending embeddings, making it prone to shifting news trends. The most effective combination, ID embeddings with projection-free self-attention, is a novel architecture in NNR literature that combines intuition from the LSTUR and NAML NNRs. Note that, by dropping the QKV projections, it restricts self-attention to leverage inter-history dependencies using semantic similarity. This is a form of regularization, and it is effective on both datasets. It also shows that self-attention in fact mostly relies on semantic similarity instead of hidden patterns uncovered through projection. The explicit regularization measures and item projection layers that we proposed, on the other hand, are only effective on NUMEND. Our regu-

larization measures were mainly designed to restrict overfitting on trends, noise and memorizing of user-item interactions, which showed to be more effective on the noisier NUMEND with fewer items, making it easier to remember, while item projection benefits from more interactions per item, which NUMEND has. On the other hand, MIND is likely more in-distribution and likely allows for more user-level optimization than NUMEND because of its more (intra-user) diverse user histories and higher quality impression negatives. So, our hypothesis that model regularization and item projection are essential for frozen NNR is rejected on MIND-small, but proven on NUMEND, and therefore possibly also the full MIND dataset. However, the bias-variance trade-off clearly shifts on both datasets: self-attention is most effective without any projections, showing that staying true to semantic space is uniquely valuable in the frozen NNR domain.

As expected, SANNR underperforms unfrozen NNR, but we found it is 100× more efficient to train and run inference. SANNR outperforms all equivalently efficient NNRs on NUMEND. However, on MIND, SANNR underperforms the Qwen3 embedding recommender. We think that such large decoder models are still not fluent in Dutch news texts, which motivates cheap domain adaptation of such embeddings through SANNR optimization. Furthermore, we argue that SANNR could still outperform Qwen3 embeddings when using it as a base news encoder, and optimizing them for user behavior that we prove it does in the next RQ.

Task adaptation characteristics Task adaptation of SANNR on NUMEND does not only cover semantic similarity of news texts: we showed that it optimizes for behavioral signal, allowing it to better predict co-occurrence probability of articles in user histories. It also reduces popularity bias relative to the semantic baseline, which suggests that even a parameter-free recommender must be task adapted to avoid reinforcing the popularity already present in user histories. Across categories, the performance of the SANNR and the semantic baseline varies: we argue that some categories on nu.nl are suited for personalized news recommendation while others are not. Categories such as 'general' cover trending *news*, and users probably do not need a personalized ranking for this category because they care about a curated set of articles that everyone else is reading on the platform. SANNR is surprisingly resilient to time lag in our tests: it shows no degradation when predicting future interests. However, the validation set seems to capture most of this short-term trending information, which the experiments do not clearly show.

User study Even though SANNR recommendations were rated significantly more precise, users rated the list of SANNR recommendations about equally relevant as the semantic recommendation list. Users seem to prefer lists that are more diverse, which SANNR lacks. However, we argue that this is still intended behavior because SANNR could add diversification later-on through re-ranking measures, which can be especially effective given the likely more distinctive rel-

evance assessments. The ranked lists were similar for most users, which further shows that SANNR maintains semantic space while slightly adapting it to make it more indicative of behavioral patterns. We think that the full labels tell the complete story, and complement our earlier findings. Apart from precision and diversity, most metrics are not statistically significant due to the small amount of users in the study.

6.2 Implications

Scientific implications We proved that there is a third way of task adaptation aside from unfreezing item encoders in NNR or fine-tuning PLM on the news domain: adding learned layers on top of PLM. Our findings suggest that further work should not treat NNR as-is when handling frozen item encoders, but treat it as a different architecture with different constraints. Because SANNR is PLM-agnostic, it connects well with recent literature about large decoder embedder models. Further scientific implications are that multilingual embedders for recommendation do not benefit from capacity, and that the architecture of NNR is highly dependent on the dataset you train it on: SANNR for instance requires more regularization and benefits from learned embedding projection. In fact, we showed that NNR architectures are very similar and can all be summarized in a taxonomy similar to SANNR defines. Also, connections to related recommender literature are often lacking, which we showed can be valuable, especially when handling frozen item encoders, which sequential recommenders commonly do.

Practical implications We showed that it is possible to efficiently adapt embeddings to a Dutch news platform so that the recommender becomes more performant and intuitive as users get recommended more similar interests and not just semantically close topics that we saw can sometimes be surprising, irrelevant, recommendations. It is even possible to do this while preserving embedding caches. This opens up many possibilities: (i) it becomes possible to add arbitrary input features because stochastic gradient descent can automatically handle them; (ii) similarly, additional mixed embedding types can be added such as video embeddings; (iii) additional loss terms can be added to automatically optimize for complex business constraints such as fairness metrics; (iv) the dataset could be bagged to train an ensemble of SANNR recommenders allowing for benefits of increased accuracy and confidence estimates; (v) the same model can be quickly re-trained for new brands without violating data privacy laws; (vi) embedding models can trivially be replaced; (vii) additional deep learning features can be added, such as multi-view article encoding, or hyper-network embedding output; (viii) it could be cheaply tested on more news platforms.

6.3 Limitations

There are limitations with how we sampled NUMEND and the features that are available. First, NUMEND is sampled from click logs instead of impression logs like MIND, which we account for with negative samples that we sample from the popularity distribution. However, these negatives may not be true negatives: users may have liked them if they would have seen them. Additionally, article popularity does not perfectly predict article exposure probability, which further introduces noise in the dataset. Second, NUMEND category labels are not perfect, and we think that some categories such as 'general' are too broad. Third, the validation set is sampled from a three days period because it would otherwise only be sampled from sunday (where news trends can be vastly different), which may be too long to measure short term interests in the test set. MIND only samples one day for this reason. Fourth, we did not have access to reading time, meaning that accidental clicks were harder to filter out, further increasing noise in the dataset.

A second limitation is that the measured performance gain of SANNR on MIND was not large compared to the semantic baselines; the Qwen3 embedding model even outperforms SANNR on this dataset. It is unclear what the effect is of the news encoder family and strategy on SANNR performance, and if SANNR achieves complementary task adaptation when combined with different news encoders such as Qwen3. Also, it is unclear if fine-tuning a news encoder on just news texts (and not on recommendation) would outperform SANNR as we did not include this baseline. Lastly, we are not sure what the performance is of end-to-end trained sentence encoders on NUMEND; this is an important reference point to find appropriate efficiency-accuracy trade-off NNR architectures.

A third limitation with our approach is that very little model capacity was required in SANNR; especially in the projection layers of self-attention where identify mapping often sufficed. Because of this, we are not sure if parameter optimization using SGD really outperforms parameter-free algorithmic methods for history encoding that augment each item in the user history with embeddings of similar content.

A fourth limitation with our approach is that we merely included one embedding per article, while it is arguably effective in NNR to include multiple views of articles such as title, body and category embeddings. This would increase the number of input features, and allow mechanisms like history encoding and fusion to aggregate from a richer pool of history embeddings. This could even be taken a step further by representing articles with n embeddings that highlight different aspects of article content.

A last major limitation is that we did not test SANNR on the joined training and validation sets of MIND and NUMEND to obtain final test performance. This is a major limitation, because our test scores now in no way indicate performance on short-term (1-3 days) user interest prediction.

6.4 Future work

Addressing both the limitations and the scientific implications, we suggest that future work could explore:

- SANNR with different base frozen news encoders to see if this also benefits from task adaptation;
- SANNR trained on the large MIND dataset so that embedding adapters optimize over more examples;
- SANNR-like frozen NNR with additional input features such as multiple views of articles;
- Parameter-free self-attention to encode user history with in SANNR;
- Shortening the validation set length to uncover short-term recommendation accuracy of SANNR;
- Ensemble methods, routing methods and additional loss terms;
- Reduce noise in NUMEND with impression logs, better category labels, and article dwell time filtering; This is only possible internally at DPG Media;
- A/B test SANNR in production.

7 Conclusions

We designed SANNR, a modular efficient neural news recommender (NNR) that explores the use of frozen news encoders, which is a novel trade-off between recommender accuracy and efficiency. We train and evaluate SANNR on MIND, and on a dataset that we sampled from nu.nl (a Dutch news platform) click logs, which we call NUMEND. Our work is split up in four parts. First, we found that news encoding on MIND benefits from large sentence encoders (or decoder models) such as Qwen3-embed while NUMEND benefits more from strong multilingual sentence encoders such as BGE-M3. Second, we identified that projection-free self-attention and user ID embeddings are effective for frozen NNR, while poly/additive attention and user-level projection is not. The projection-free self-attention results highlight that frozen NNR heavily rely on semantic space for self-attention. Furthermore, we found that item-level embedding projection and explicit regularization measures are not effective on MIND but are effective on NUMEND. The best SANNR configurations improve recommender ranking performance on MIND and NUMEND over a semantic baseline by about 10% in our offline evaluation. Additionally, SANNR also outperforms NRMS, LSTUR, MINER, and NAML with frozen news encoders, and the specialized NaSE news encoder for recommendation on ranking and classification performance, while remaining equally efficient. End-to-end

trained NNR outperform SANNR by about 15% nDCG@10 ranking performance, but require 100x the training/inference compute. In the third part of our research, we identified that SANNR optimizes for user behavior, detrends popularity signal, works for any user history length, generalizes to future user interactions, and is able to optimize well for at least some news categories. In the last part of the project, we verified SANNR recommender performance in a user study, where we confirm that SANNR recommendations are about 30% more precise but also less diverse than a late fusion semantic recommender.

All in all, our work shows that frozen NNR can achieve task adaptation for recommendation in a very efficient, practical manner, and that frozen NNR benefit from embedding projection, user modeling components such as self-attention and ID embeddings, and regularization measures which is different from end-to-end trained NNR. Furthermore, the item interaction sparsity and level of noise in the dataset also plays a large role in designing (frozen) NNR.

Acknowledgements

I want to thank Suzan Verberne and Marlies van der Wees for the great advice and collaboration throughout this project. Suzan, you really helped me with keeping analyses simple and true to their purpose of gaining insights about something, structuring my thesis, great feedback on my writing, better understanding core topics, suggestions for research direction, pulling me out of some rabbit holes, and occasionally with some positive reinforcement that I was on the right path that I became blind to. I really enjoyed our bi-weekly meetings where we could talk about the project and just about how everything was going. Marlies, thank you very much for the opportunity that I could do the internship in your team under your supervision. I really enjoyed my time at DPG Media and our coffee breaks, team building activities and weekly meetings. You also really helped me a lot with the thesis. For instance, I found it very helpful that you can glance at some results that I had looked at for hours and point out something I had missed. Furthermore, I really valued your guidance on who to talk to within DPG, learning about in-production challenges, and valuable research directions. Also, I really liked how you were always able to share your passion and excitement for recommender systems. At DPG Media, I also want to thank Robin Verachtert, Kim Falk, and Rosa Maria Siervo because they all helped with the project with their input.

References

Abdollahpouri et al. (June 21, 2021). "User-Centered Evaluation of Popularity Bias in Recommender Systems". In: *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization*. UMAP '21. New York, NY, USA: Association for Computing Machinery, pp. 119–129. ISBN: 978-1-4503-8366-0. DOI: 10.1145/3450613.3456821. URL: https://dl.acm.org/doi/10.1145/3450613.3456821 (visited on 11/06/2025).

An et al. (July 2019). "LSTUR: Neural News Recommendation with Long- and Short-term User Representations". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. ACL 2019. Ed. by Anna Korhonen, David Traum, and Lluís Màrquez. Florence, Italy: Association for Computational Linguistics, pp. 336–345. DOI: 10.18653/v1/P19-1033. URL: https://aclanthology.org/P19-1033/ (visited on 09/12/2025).

BaKirosHinton (2016). "Layer normalization". In: *arXiv preprint arXiv:1607.06450*.

Chen et al. (July 7, 2022). "Generative Adversarial Framework for Cold-Start Item Recommendation". In: *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '22. New York, NY, USA: Association for Computing Machinery, pp. 2565–2571. ISBN: 978-1-4503-8732-3. DOI: 10.1145/3477495.3531897. URL: https://dl.acm.org/doi/10.1145/3477495.3531897 (visited on 03/03/2026).

Al-Hassan et al. (Mar. 2024). "An Improved Fusion-Based Semantic Similarity Measure for Effective Collaborative Filtering Recommendations". In: *International Journal of Computational Intelligence Systems* 17.1, p. 45. ISSN: 1875-6883. DOI: 10.1007/s44196-024-00429-4. (Visited on 12/12/2025).

He et al. (2017). "Neural collaborative filtering". In: *Proceedings of the 26th international conference on world wide web*, pp. 173–182.

HendrycksGimpel (2016). "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415*.

Hu et al. (July 13, 2025). "AlphaFuse: Learn ID Embeddings for Sequential Recommendation in Null Space of Language Embeddings". In: *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '25. New York, NY, USA: Association for Computing Machinery, pp. 1614–1623. ISBN: 979-8-4007-1592-1. DOI: 10.1145/3726302.3729894. URL: https://dl.acm.org/doi/10.1145/3726302.3729894 (visited on 03/03/2026).

Huang et al. (July 18, 2023). "Aligning Distillation For Cold-start Item Recommendation". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '23. New York, NY, USA: Association for Computing Machinery, pp. 1147–1157. ISBN: 978-1-4503-9408-6. DOI: 10.1145/3539618.3591732. URL: https://dl.acm.org/doi/10.1145/3539618.3591732 (visited on 03/04/2026).

IanaGlavasPaulheim (July 18, 2023). "Simplifying Content-Based Neural News Recommendation: On User Modeling and Training Objectives". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and De-*

velopment in Information Retrieval. SIGIR '23. New York, NY, USA: Association for Computing Machinery, pp. 2384–2388. ISBN: 978-1-4503-9408-6. DOI: 10.1145/3539618.3592062. URL: https://dl.acm.org/doi/10.1145/3539618.3592062 (visited on 02/27/2026).

Iana et al. (2025). "[NaSE] News Without Borders: Domain Adaptation of Multilingual Sentence Embeddings for Cross-Lingual News Recommendation". In: *Advances in Information Retrieval*. Ed. by Claudia Hauff et al. Cham: Springer Nature Switzerland, pp. 119–137. ISBN: 978-3-031-88711-6. DOI: 10.1007/978-3-031-88711-6_8.

Jain et al. (2023). *NEFTune: Noisy Embeddings Improve Instruction Finetuning*. Tech. rep. Lawrence Livermore National Laboratory (LLNL), Livermore, CA (United States).

Li et al. (May 2022). "MINER: Multi-Interest Matching Network for News Recommendation". In: *Findings of the Association for Computational Linguistics: ACL 2022*. Findings 2022. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Dublin, Ireland: Association for Computational Linguistics, pp. 343–352. DOI: 10.18653/v1/2022.findings-acl.29. URL: https://aclanthology.org/2022.findings-acl.29/ (visited on 01/07/2026).

Li et al. (Nov. 10, 2025). "Exploring the Upper Limits of Text-Based Collaborative Filtering Using Large Language Models: Discoveries and Insights". In: *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*. CIKM '25. New York, NY, USA: Association for Computing Machinery, pp. 1643–1653. ISBN: 979-8-4007-2040-6. DOI: 10.1145/3746252.3761429. URL: https://dl.acm.org/doi/10.1145/3746252.3761429 (visited on 02/27/2026).

LiuDolanPedersen (Feb. 7, 2010). "Personalized News Recommendation Based on Click Behavior". In: *Proceedings of the 15th International Conference on Intelligent User Interfaces*. IUI '10: 15th International Conference on Intelligent User Interfaces. Hong Kong China: ACM, pp. 31–40. ISBN: 978-1-60558-515-4. DOI: 10.1145/1719970.1719976. URL: https://dl.acm.org/doi/10.1145/1719970.1719976 (visited on 11/10/2025).

Lu (2020). *Bag of Tricks and a Strong Baseline for Neural News Recommendation*. MIND News Recommendation Competition technical report.

Meng et al. (Dec. 1, 2023). "A Survey of Personalized News Recommendation". In: *Data Science and Engineering* 8.4, pp. 396–416. ISSN: 2364-1541. DOI: 10.1007/s41019-023-00228-5. URL: https://doi.org/10.1007/s41019-023-00228-5 (visited on 09/02/2025).

MiyatoDaiGoodfellow (2017). "ADVERSARIAL TRAINING METHODS FOR SEMI-SUPERVISED TEXT CLASSIFICATION". In: *stat* 1050, p. 6.

OrtakciBorhan (2025). "Optimizing SBERT for long text clustering: two novel approaches with empirical insights: Y. Ortakci, B. Borhan". In: *The Journal of Supercomputing* 81.8, p. 950.

Papadakis et al. (Jan. 1, 2022). "Collaborative filtering recommender systems taxonomy". In: *Knowledge and Information Systems* 64.1, pp. 35–74. ISSN: 0219-3116. DOI: 10.1007/s10115-021-01628-7. URL: https://doi.org/10.1007/s10115-021-01628-7 (visited on 06/04/2026).

Paterek (2007). “Improving regularized singular value decomposition for collaborative filtering”. In: *Proceedings of KDD cup and workshop*. Vol. 2007. 2007, pp. 5–8.

Pembek et al. (Sept. 7, 2025). “Let It Go? Not Quite: Addressing Item Cold Start in Sequential Recommendations with Content-Based Initialization”. In: *Proceedings of the Nineteenth ACM Conference on Recommender Systems*. RecSys ’25. New York, NY, USA: Association for Computing Machinery, pp. 626–631. ISBN: 979-8-4007-1364-4. DOI: 10.1145/3705328.3748038. URL: https://dl.acm.org/doi/10.1145/3705328.3748038 (visited on 03/03/2026).

Qi et al. (July 6, 2022). “FUM: Fine-grained and Fast User Modeling for News Recommendation”. In: *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. Madrid Spain: ACM, pp. 1974–1978. ISBN: 978-1-4503-8732-3. DOI: 10.1145/3477495.3531790. URL: https://dl.acm.org/doi/10.1145/3477495.3531790 (visited on 09/12/2025).

Rajput et al. (Dec. 15, 2023). “Recommender Systems with Generative Retrieval”. In: *Advances in Neural Information Processing Systems* 36, pp. 10299–10315. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/20dcab0f14046a5c6b02b61da9f13229-Abstract-Conference.html (visited on 03/10/2026).

Schafer et al. (2007). “Collaborative filtering recommender systems”. In: *The adaptive web: methods and strategies of web personalization*. Springer, pp. 291–324.

Song et al. (2020). “Mpnet: Masked and permuted pre-training for language understanding”. In: *Advances in neural information processing systems* 33, pp. 16857–16867.

TakácsTikk (2012). “Alternating least squares for personalized ranking”. In: *Proceedings of the sixth ACM conference on Recommender systems*, pp. 83–90.

VolkovsYuPoutanen (2017). “DropoutNet: Addressing Cold Start in Recommender Systems”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/dbd22ba3bd0df8f385bdac3e9f8be207-Abstract.html (visited on 03/03/2026).

Wu et al. (Aug. 2019a). “NAML: Neural News Recommendation with Attentive Multi-View Learning”. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. Macao, China: International Joint Conferences on Artificial Intelligence Organization, pp. 3863–3869. ISBN: 978-0-9992411-4-1. DOI: 10.24963/ijcai.2019/536. (Visited on 01/07/2026).

– (July 25, 2019b). “NPA: Neural News Recommendation with Personalized Attention”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD ’19. New York, NY, USA: Association for Computing Machinery, pp. 2576–2584. ISBN: 978-1-4503-6201-6. DOI: 10.1145/3292500.3330665. URL: https://doi.org/10.1145/3292500.3330665 (visited on 01/07/2026).

Wu et al. (Nov. 2019c). "NRMS: Neural News Recommendation with Multi-Head Self-Attention". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. EMNLP-IJCNLP 2019. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, pp. 6389–6394. DOI: 10.18653/v1/D19-1671. URL: https://aclanthology.org/D19-1671/ (visited on 09/11/2025).

Wu et al. (July 11, 2021). "Empowering News Recommendation with Pre-trained Language Models". In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '21. New York, NY, USA: Association for Computing Machinery, pp. 1652–1656. ISBN: 978-1-4503-8037-9. DOI: 10.1145/3404835.3463069. URL: https://doi.org/10.1145/3404835.3463069 (visited on 01/07/2026).

Wu et al. (Jan. 2023). "Personalized News Recommendation: Methods and Challenges". In: *ACM Trans. Inf. Syst.* 41.1, 24:1–24:50. ISSN: 1046-8188. DOI: 10.1145/3530257. (Visited on 09/02/2025).

Wu et al. (2020). "Mind: A large-scale dataset for news recommendation". In: *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 3597–3606.

Xiao et al. (Aug. 14, 2022). "Training Large-Scale News Recommenders with Pretrained Language Models in the Loop". In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD '22. New York, NY, USA: Association for Computing Machinery, pp. 4215–4225. ISBN: 978-1-4503-9385-0. DOI: 10.1145/3534678.3539120. URL: https://dl.acm.org/doi/10.1145/3534678.3539120 (visited on 02/27/2026).

Yuan et al. (July 2023). "Where to Go Next for Recommender Systems? ID- vs. Modality-based Recommender Models Revisited". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '23. New York, NY, USA: Association for Computing Machinery, pp. 2639–2649. ISBN: 978-1-4503-9408-6. DOI: 10.1145/3539618.3591932. (Visited on 01/07/2026).

Zhang et al. (May 2024). "Are ID Embeddings Necessary? Whitening Pre-trained Text Embeddings for Effective Sequential Recommendation". In: *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. 2024 IEEE 40th International Conference on Data Engineering (ICDE), pp. 530–543. DOI: 10.1109/ICDE60146.2024.00063. URL: https://ieeexplore.ieee.org/abstract/document/10598127 (visited on 03/03/2026).

Zhang et al. (2005). "Using singular value decomposition approximation for collaborative filtering". In: *Seventh IEEE International Conference on E-Commerce Technology (CEC'05)*. IEEE, pp. 257–264.

Zheng et al. (2021). "Disentangling user interest and conformity for recommendation with causal embedding". In: *Proceedings of the web conference 2021*, pp. 2980–2991.