

AI-powered News Recommender System

Wyatt Meng
2024 May 22th

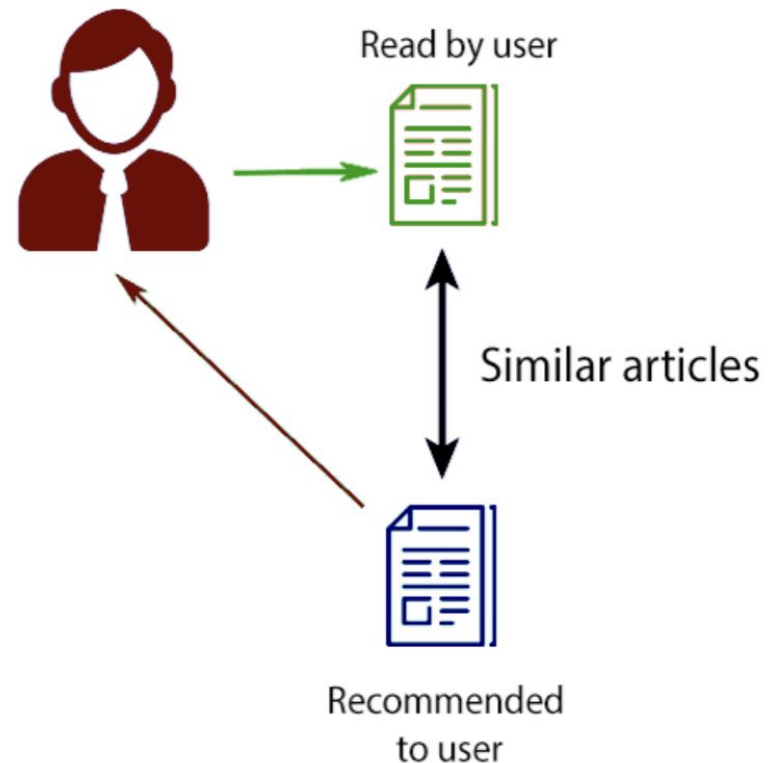
Problem Statement

Problem: Online news platforms aggregate content from numerous sources, presenting a vast array of articles. We want to build a recommender system to find news that resonates with user specific interests. Traditional systems often fall short due to the **dynamic nature of news** relevancy and the **unique interest and information needs of each user**

An Ideal Content-based Recommender System:

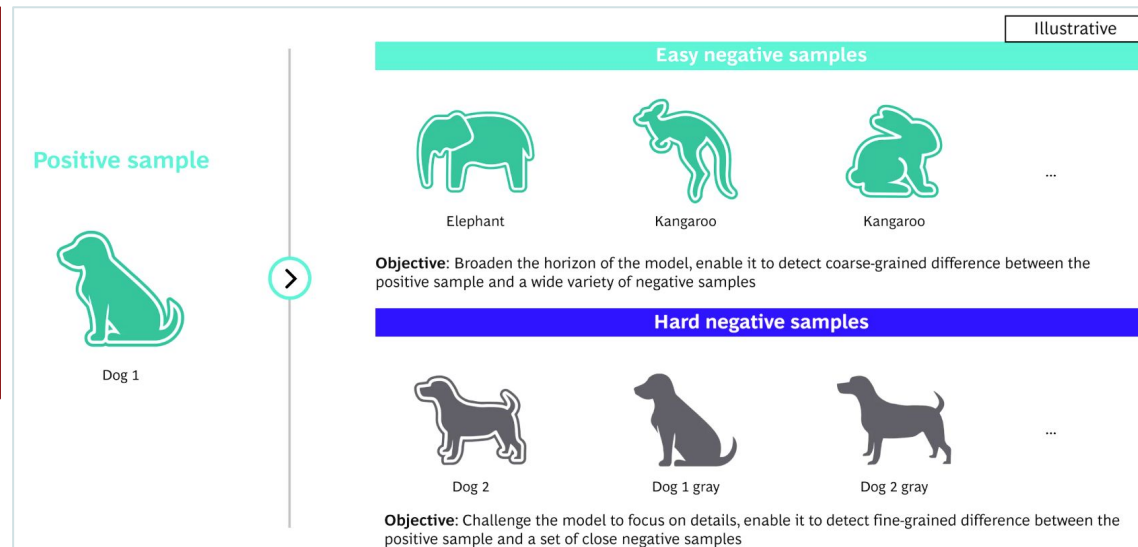
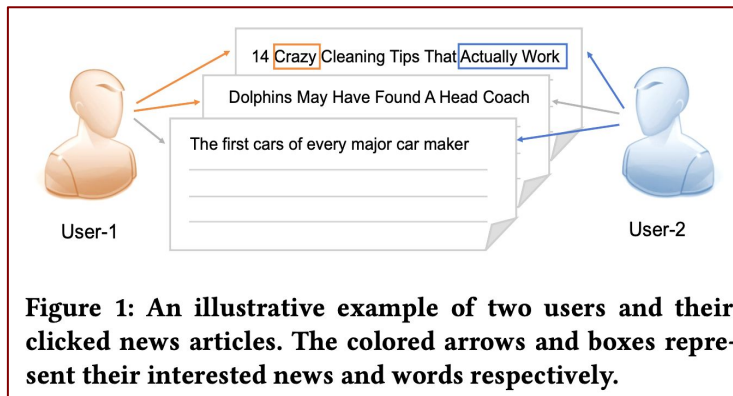
Maximize Engagement: High engagement measured by metrics like Normalized Discounted Cumulative Gain (NDCG) and Area Under Curve (AUC)

Increase Content Diversity: Maintain diversity in recommendations to ensure users are exposed to a wide range of news topics and sources.



Assumptions/Hypotheses about data and model

1. **Data Representation:** User interaction data (clicks, impressions) is representative of user preferences; Our labels are accurate, so we can use a supervised framework
2. **Data Quality:** Structured datasets where the format and integrity of data are controlled; thus, specific handling of duplication, missing values or outliers is not necessary
3. **Negative Sampling Adequacy:** Negative sampling (generating non-clicked impressions as negative examples) approximates the full range of negative user interactions. This is important for the model to learn discrimination between liked and disliked news items
4. **Embedding Effectiveness:** Both models hypothesize that embeddings (word, entity, and user) capture semantic and contextual information that can predict user preferences
5. **Historical Relevance:** past user interactions (click history) are indicative of future interests, which is fundamental for the attention mechanisms in both models to function



Exploratory Data Analysis I

Overview of Data: The MINDsmall dataset includes separate *training*, *validation*, and *testing* subsets, each with dedicated **news** and **behaviors** files, organized for *recommender systems*

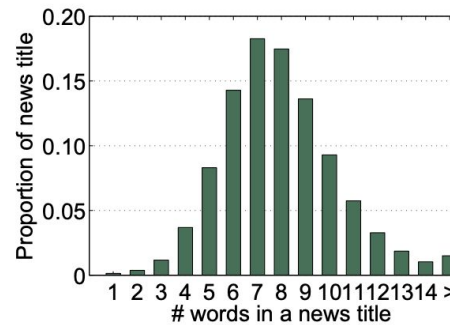
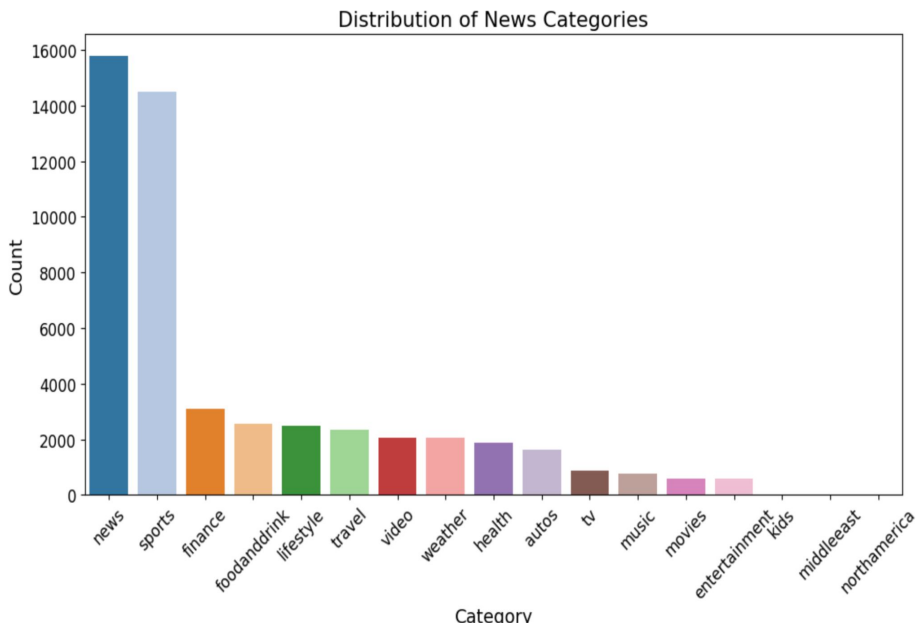
Training: 51,282 news entries and 156,965 behavior logs

Validation: Comprises 42,416 news entries and 73,152 behavior logs.

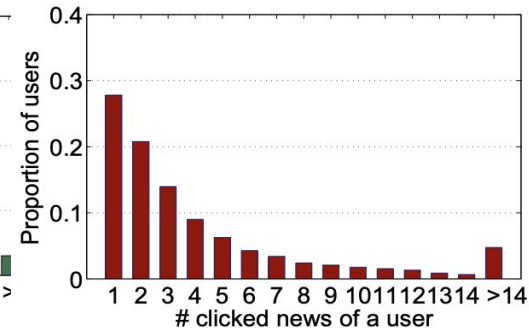
News: aimed at item profiling, featuring detailed attributes like news ID, category, title, abstract, and entities; forming the basis for content-based filtering techniques.

Behaviors: captures user interactions, including user ID, click history, and impressions, essential for learning user preferences and implementing collaborative filtering approaches.

Missing Data Analysis: Training set has 2,666 and 3,238 missing for news abstracts and User click history respectively. Validation data shows similar trends with 2,021 missing news abstracts and 2,214 missing user histories.



Distribution of the number of clicked news of a user



Distribution of the number of words in a news title

Exploratory Data Analysis II: Sentiment Analysis using Distilled Bert

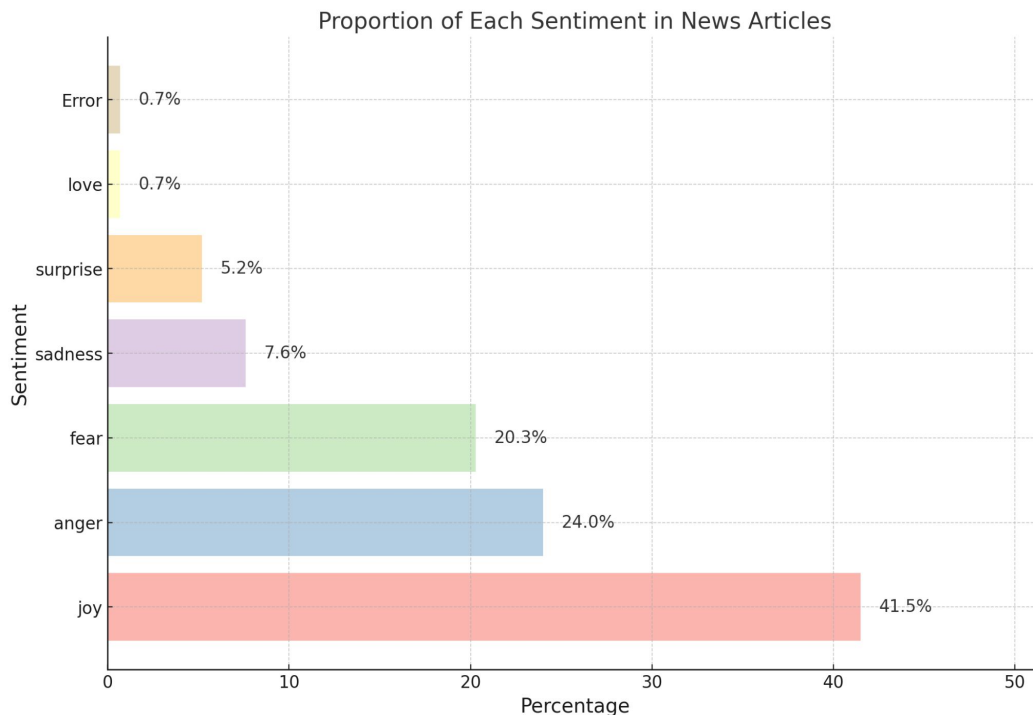
Pipeline Initialization: Utilizes `transformers` library to implement a text-classification pipeline with the pre-trained model '[/distilbert-base-uncased-emotion](#)'.

Model Efficiency: Utilizes `DistilBERT`, which is 40% smaller but retains 97% of its language understanding capabilities comparing to BERT

Emotion Detection Function: `detect_mood` that analyzes text to identify emotions. The function captures and returns the most probable sentiment label or 'Error' in case of an exception.

Sentiment Analysis Execution: Applies the `detect_mood` function to a random sample of 10 news abstracts from the `train_news` dataset, enriching the dataset with mood labels.

Appended as a new column for better predictive results



Feature Engineering & Transformations

Session Parsing: Extracted user interaction training and validation data, including clicked and not-clicked news items from user click history logs

Negative Sampling: Applied negative sampling techniques to generate synthetic negative samples, enhancing the dataset for more robust training

Text Tokenization: Utilized **RegexTokenizer** to break down news titles into words, preparing textual content

Model 1 DKN: Integrates knowledge graph embeddings to enrich news representations with semantic and relational data.

Employs a knowledge-aware convolutional neural network (KCNN) that aligns word and entity embeddings.

Model 1 NPA: Used CNNs for transforming news titles into informative vector embeddings.



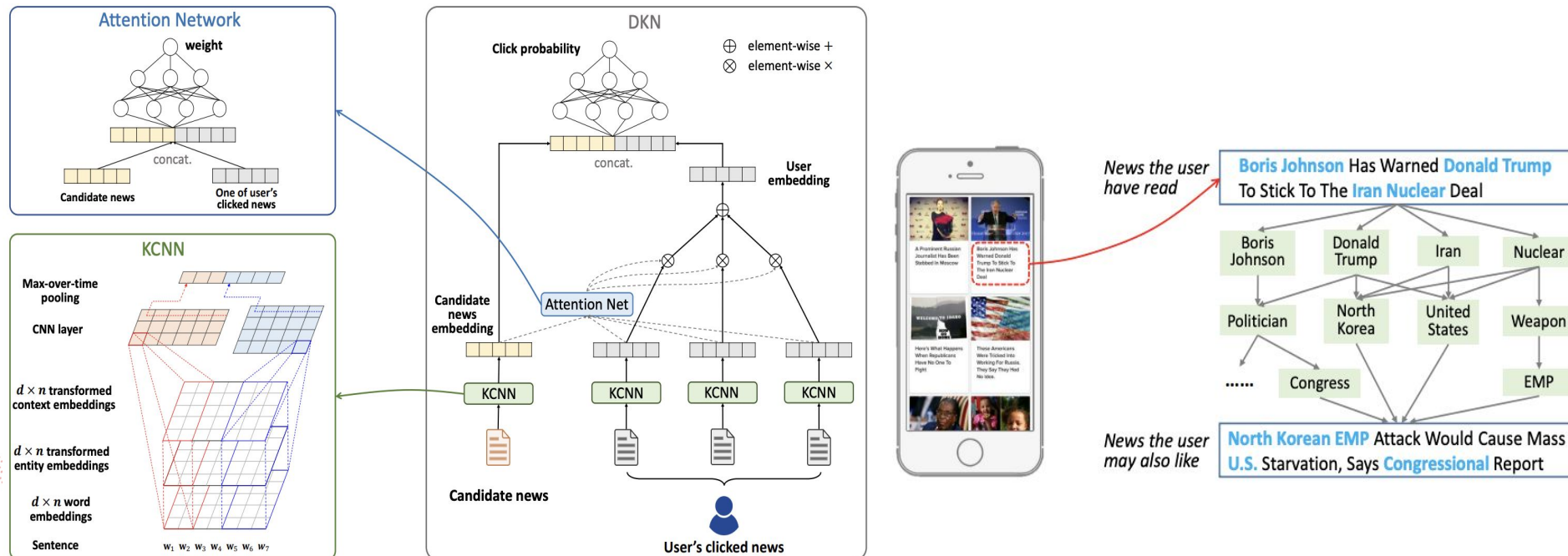
Recommender System(RS) Model I:

DKN : Deep Knowledge-Aware Network for News Recommendation

Model Focus: Content-based RS, Integrates **knowledge graph(KG)** with TransX method to enrich the recommendation process, accounting for external semantic relationships.

Knowledge-aware Convolutional Neural Network (KCNN) : KCNN fuses word embeddings with entity embeddings from a KG, treating **words and associated entities as multiple channels in the convolution process**.

User Modeling: Incorporates an attention to aggregates a user's interactions with news, measuring the relevance of current candidate news based on past user activity.



Recommender System(RS) Model II:

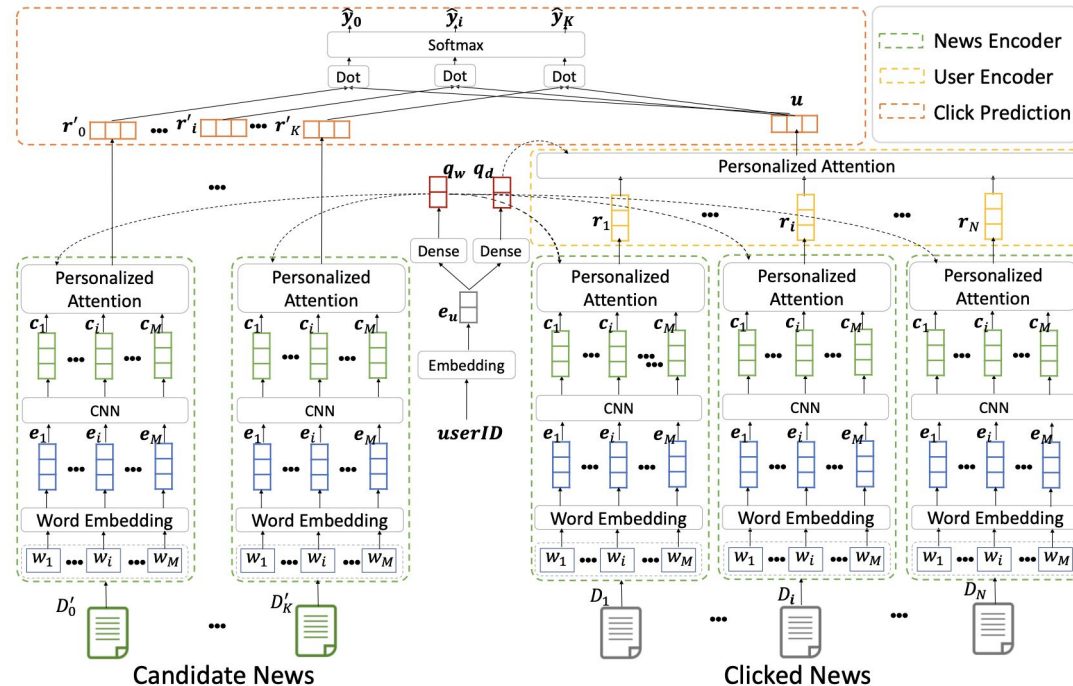
NPA: Neural News Recommendation with Personalized Attention

NPA: a **content-based** news RS

Uses a **CNN** network to learn news representation. Learns user representations from their **clicked news articles**

A **word-level personalized attention** is used to help NPA attend to important words for different users

A news-level personalized attention is used to help NPA attend to important historical clicked news for different users.



Three major modules:

1. **News encoder:** learns new representations
2. **User encoder:** learns user representations based on the clicked news(impressions)
3. **Click predictor:** predict the click score of a series of candidate news.

Recommender System(RS) Model checks for overfitting/underfitting

NPA : 1. **Dropout Strategy**: utilizes dropout as a regularization technique in each layer of the network. The **dropout rate was set to 0.2**, preventing overfitting by randomly dropping units (along with their connections) during the training process. This ensures that the network does not rely on any specific set of features, promoting generalization.

2. **Repeated Experiments**: Original paper conducted each **experiment 10 times** and reporting **average metrics (AUC, MRR, nDCG@5, nDCG@10)** further validates the stability and reliability of the model under different initializations and splits of data.

DKN : 1. **Parameter Sensitivity Analysis**: examines the **sensitivity of the model to hyperparameters** like the dimension of word and entity embeddings and the configuration of the convolutional filters (number and size). This analysis helps in understanding the impact of these parameters on performance and guards against overfitting by **identifying when increasing complexity** (e.g., too many filters or too large embeddings) starts to hurt performance.

2. **Effect of Window Sizes and Number of Filters**: The study on how changing the window sizes and the number of filters impacts the model helps in tuning these parameters to optimize performance without overfitting. As noted, **very large window sizes or too many filters can lead to overfitting, recognizing** and mitigating which is crucial for maintaining a robust model.



Recommender System(RS) Model Result Comparative Analysis I

4 Evaluation metrics

AUC: Quantifies model's ability to distinguish classes;

Mean Reciprocal Rank(MRR): Mean of inverse ranks of the first relevant item.

Normalized Discounted Cumulative Gain (nDCG@5 & nDCG@10): Measures rank quality, how well the predicted items for a user are ranked based on relevance; penalizes lower-position items.

$$nDCG = \frac{DCG}{IDCG}, \text{ where } DCG = \sum_{i=1}^p \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}$$

Test set: Time period includes user logs from the last week of a one-month period (from December 13, 2018, to January 12, 2019). The test set is randomly sampling from MSN; News user behavior and interactions. We defined a function for each metric above and calculated these metrics, shown below:

AUC : DKN: 0.5869, indicates **moderate discrimination** between relevant and non-relevant news.

NPA: 0.6243*, demonstrates **better discrimination capabilities**

MRR: DKN: MRR of 0.3044 indicates relevant results are generally near the top.

NPA: Improved MRR of 0.3321, more consistently **places relevant news at the highest ranks.**

nDCG@5: DKN: 0.3184, fair early ranking of relevant news.

NPA: 0.3535*, **better at ranking highly relevant news** within the top 5 positions.

Methods	AUC	MRR	nDCG@5	nDCG@10
DKN	0.5869	0.3044	0.3184	0.4071
NPA	0.6243*	0.3321*	0.3535*	0.4380*



Recommender System(RS) Model Result Comparative Analysis II

NPA has a better result

Attention Mechanisms: Both **DKN** and **NPA** used attention but **NPA** focuses on personalizing this attention at both the word and news article levels based on user-specific embeddings

DKN uses attention to dynamically **integrate user history with current news recommendations**, with a strong emphasis on semantic, entity relationship and knowledge graph information.

Knowledge Integration: **DKN** stands out by integrating knowledge graphs to understand and link news content at a deeper semantic level, a feature that NPA does not possess.

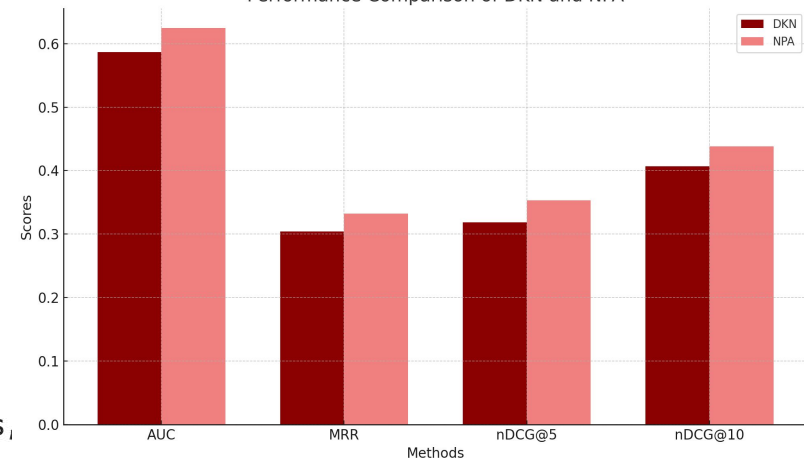
Purpose of Modeling: **NPA** aim to enhance personalization in news recommendations by **adapting its responses based on the individual's past interactions**

DKN aims to **broaden the recommendation scope** by incorporating external knowledge, thereby potentially **discovering latent connections** among news topics

```
%time
model.fit(train_news_file, train_behaviors_file, valid_news_file, valid_behaviors_file)

... 1086it [35:31, 1.96s/it]
8874it [44:08, 3.35it/s]
at epoch 1
train info: logloss loss:1.521172252149213
eval info: group_auc:0.5814, mean_mrr:0.2514, ndcg@10:0.3375, ndcg@5:0.2691
at epoch 1, train time: 2131.3 eval time: 2658.9
743it [23:55, 1.92s/it]
```

Performance Comparison of DKN and NPA



```
updates=self.state_updates,
8874it [46:00, 3.22it/s]
{'group_auc': 0.5053, 'mean_mrr': 0.22, 'ndcg@5': 0.2236, 'ndcg@10': 0.2902}
```

THE UNIVERSITY OF CHICAGO

DATA SCIENCE
INSTITUTE

Future Work

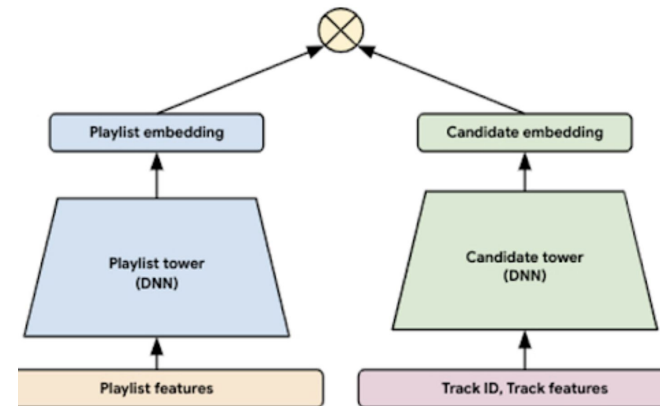
Two-Tower Model Implementation: Adopt a Two-Tower model to simultaneously process user and item data, enhancing recommendation precision by leveraging dual encoders for embedding.

Advanced Contextual Understanding: Integrate deeper contextual analysis and sentiment understanding using NLP models

We used BERT to perform sentiment analysis, we added this to the current system but did not see a significant improvement in metrics

Real-Time User Feedback Integration: Develop real-time feedback loops using reinforcement learning to refine user profiles and recommendation accuracy

Dynamic Knowledge Graphs: Maintain dynamic knowledge graphs that evolve with news trends and user interactions to improve recommendation relevance.



Appendix

Two-Tower Model

The Two-Tower model (also known as the Dual-Encoder or Siamese Network model) is a deep learning-based architecture commonly used in large-scale recommendation systems. It consists of two separate neural networks (towers) that process different types of input—typically user data and item data—and project them into a shared embedding space. The similarity between the user and item embeddings is then used to make recommendations.

User Tower: Processes user-related features such as user demographics and interaction history. The user tower transforms these features into user embeddings that represent the user's preferences in the embedding space.

Item Tower: Processes item-related features such as content metadata and item embeddings. The item tower converts these features into item embeddings that represent the characteristics of the items.

<https://cloud.google.com/blog/products/ai-machine-learning/scaling-deep-retrieval-tensorflow-two-towers-architecture>





<https://towardsdatascience.com/two-tower-networks-and-negative-sampling-in-recommender-systems-fdc88411601b>



Visualization of the attention weights from the word- and news-level personalized attention network. The users and news are randomly sampled from the dataset. Darker colors indicate higher attention weights.

	User 1	User 2
Clicked	' no doubt ' kyler murray loves football more than baseball	holiday movie guide 2018 : every movie you should see
News	warriors by far most hated nba team in state-by-state survey	celebrate the holidays with your favorite tv shows
	the most famous actor the same age as you	spider-man : into the spider-verse ' swings to top box office spot
	winners and losers from nfl week 17	the most famous actor the same age as you
Candidate	5 most desirable super bowl matchups	5 most desirable super bowl matchups
News	year of the superheroes : the highest-grossing movies of 2018	year of the superheroes : the highest-grossing movies of 2018

(a) Word-level attention weights.

	User 1	User 2
	 10 nfl coaches who could be fired on black monday	 the news in cartoons
	 nfl releases playoff schedule : cowboys-seahawks on saturday night	 10 nfl coaches who could be fired on black monday
	 raiders ' harris scores on insane 99-yard punt return	 spider-man : into the spider-verse ' swings to top box office spot
	 new year 's eve weather forecast	 year in review : 100 best movies of 2018
	 best photos of 2018	 best photos of 2018
	 stunning photos show what holidays look like around the world	 up to 6 inches of snow piles up in west texas

(b) News-level attention weights.

