Developing Generative Recommender Systems for Government Subsidy Programs with a New RQ-VAE Model: Wello & the Korean Government Case

Ji Won Kim[1], Jae Hong Park[1], Yuri Anna Kim[2], Sang Jun Lee[2]

[1] Kyung Hee University

[2] Wello Inc

won103203@khu.ac.kr, jaehp@khu.ac.kr, yuri@wello.info, kevin@wello.info

Abstract

According to an industry survey, many people miss opportunities to apply for government subsidy programs because they do not know how to apply. People also need to search manually and check whether these programs are suitable for them. To address this issue, our study develops a new generative recommender system with both users' information and government subsidy documents. Within our recommender system framework, we modify the existing Residual Quantization Variational Auto-Encoder (RQ-VAE) model to capture deep and abstract information from subsidy documents. Using semantic IDs generated for approximately 185,610 user click-stream histories and 240,000 documents, we train our recommender system to predict the semantic IDs of the next subsidy policy documents in which a user might be interested. In 2024, we successfully deploy our generative recommender system in Wello, a Korean Gov-Tech startup. In collaboration with the Korean government, our generative recommender system could save $7.8 million, that might otherwise have gone unused due to a lack of applications. Also, Wello observed a 68% improvement in Click-Through Ratio (CTR), increasing from 41.4% in the third quarter of 2024 to 69.6% in the fourth quarter of 2024. We thus anticipate that our generative recommender system will have a significant impact on both individuals and the government.

Introduction

According to an industry survey (Embrain 2023), 55 percent of people lost opportunities to apply for government subsidy programs because they did not know how to apply for them. Also, many people waste time searching for government subsidy programs because such information is often spread across several websites, and they have to check whether each program is a good fit with them. Government agencies in South Korea report that they spend more than $950 million for advertising their subsidy programs every year. However, they also report that the allocated $35.8 billion for subsidy programs remained unspent. Of this amount, $2.7 billion was allocated but left unused due to a lack of applications. In response, the Korean government sought a third-party firm to manage all the subsidy information in one place. Accordingly, Wello filled this role and became one of the biggest government tech firms in South Korea.

With the cooperation of the Korean government, Wello has aggregated and promoted all government subsidy programs since 2020. As of the start of 2024, Wello had more than 230,000 active users and promoted around 240,000 government subsidy documents on their platform. With this volume of documents, users at Wello still needed to spend time and effort to find a particular subsidy program. Although Wello provided a simple recommendation based on keywords (i.e., the time left to apply and benefit amount)[1], many users still had a hard time finding the right subsidy programs, which might eventually lead to a lack of applications. To address this, we developed a novel AI-based recommender system, simply called the generative recommender system, to recommend relevant subsidy programs accurately to a particular user. We believe that our generative recommender system dramatically improves Wello's performance and saves the Korean government money for subsidy programs by helping more people find relevant subsidy programs.

We developed a new recommender system based on the generative retrieval technique, which directly generates candidates without a ranking model and vector search methods (Jin 2024). Existing generative recommendation techniques encode item features into semantic IDs by using a Vector Quantization Variational Auto-Encoder (VQ-VAE) (Van and Vinyals 2017) or Residual Quantization Variational

[1] The performance of past recommendation algorithm at Wello platform was not good enough. For instance, young males sometimes encountered the subsidy program "Financial Help for Pregnant Women" in their recommendation slot.

Auto-Encoder (RQ-VAE) (Zeghidour et al. 2021, Adiban et al. 2022). Through these semantic IDs, the generative recommender system can understand the semantic meaning of an item (Sun et al. 2024). However, because the contents of government subsidy documents are sometimes highly similar to each other, the existing generative recommender system might have difficulty distinguishing deep semantic differences among similar subsidy documents. Thus, we decided to develop a new RQ-VAE model that captures semantically deeper and more abstract information (i.e., subsidy subjects or topics) from similar documents. Specifically, in the generative retrieval recommendation architecture, we noticed that the existing RQ-VAE model tends to primarily capture the most important information at the first codebook (Zheng et al. 2024). To address this, we modified the existing RQ-VAE model with the features of VQ-VAE models and let each module learn patterns of residual representations. Our new RQ-VAE model can thus understand deeper and more abstract information.

In this paper, we use the Wello dataset to develop our generative recommender system. This dataset includes 185,610 user click-stream histories (e.g., demographic information, personal preferences, and "like" or "wish" logs on the subsidy programs) and over 240,000 subsidy documents. We demonstrate that our best model achieves performance comparable to previous studies (Rajput et al. 2024), with results of 0.4084 for Recall@5, 0.2964 for NDCG@5, 0.5017 for Recall@10, and 0.3264 for NDCG@10. Then, we began to deploy our Generative Recommender System on the Wello platform in the second quarter of 2024. The Wello technical team continuously updates and optimizes our generative recommender system monthly with new government subsidy documents and user clickstream histories.

This study, then, provides the academic contribution to related literature. Unlike previous studies in generative retrieval recommender systems utilizing item (or document) only (Rajput et al. 2024, Jin et al. 2024), our study is the first to use both subsidy documents and user preference information. In addition, we modify the existing RQ-VAE model to extract deep and abstract information from government subsidy documents correctly. We also observe that our generative recommender system shows great performance as compared to previous studies (Hidasi et al. 2015, Kang and McAuley 2018, Sun et al. 2019, Tay et al. 2021, Geng et al. 2022, Rajput et al. 2024).

Our study also provides several contributions to the government tech industry. We are the first to construct a subsidy program document database in South Korea with users' preference and demographic information. Furthermore, we complete to deploy an AI-based generative recommender system on the Wello platform in the third quarter of 2024. According to our A/B tests, Wello has observed a 68% improvement in Click-Through Ratio (CTR), increasing from 41.4% in the third quarter of 2024 to 69.6% in the fourth quarter of 2024. Also, from a survey by Wello, we can increase users' satisfaction with the new recommender system by about 19.8% in the same period.

More importantly, we observed that 36% more people applied for government subsidy programs on the platform. By doing so, the Korean government expects to save $7.8 million that might otherwise have gone unused due to a lack of applications. Also, we estimate that our new recommender system could save around $27.4 million in advertising subsidy policy programs. So, we believe that the effects of our generative recommender systems for government subsidy programs will grow substantially by the end of 2024. We observed that the total benefit amount of subsidy programs that users viewed on the platform increased by 46.7%, from $1900 in the third quarter of 2024 to $2740 in the fourth quarter of 2024.

Related Works

Vector Quantization VAE

A Variational Auto-Encoder (VAE) model compresses original data into a lower-dimensional space using an encoder while capturing patterns to enable the decoder to reconstruct the original data (Van and Vinyals 2017). However, vector quantization often leads to data distortion and reconstruction issues in VQ-VAE. To address this, the Residual Quantization VAE (RQ-VAE) was developed, utilizing multiple codebooks (Zeghidour et al. 2021, Adiban et al. 2022). Moreover, recent research indicates that RQ-VAE faces an imbalance problem among codebooks (Zheng et al. 2024). We hence propose to insert encoder-decoder modules between the codebooks. This approach allows the model to mitigate imbalance problems, as the encoder can capture important features and discard unnecessary information during reconstruction (Bengio et al., 2013).

Generative Retrieval Techniques

Traditional document retrieval techniques use vector similarity computations, which become computationally expensive as the database size increases. Traditional techniques also face challenges in computation since they have a large embedding table (Lee et al. 2022). Generative retrieval is a recent approach that aims to fix such issues by producing document IDs or document titles respectively. DSI (Tay et al. 2022) was the first approach to retrieve entities by generating their unique names using a transformer architecture. Similarly, GENRE (Sun et al. 2024) employs transformer architecture for information retrieval, but it goes further by encoding not only documents but also queries to sequentially decode relevant document IDs. We therefore use generative retrieval techniques in our recommender systems to

generate the next subsidy program document's IDs token by token.

Recommender Systems and Deployment

Recent studies of sequential recommender systems have focused on leveraging sequences of user histories. GRU4REC (Hidasi et al. 2015) was one of the first models to utilize long user histories, addressing the limitations of matrix factorization approaches. With the introduction of the self-attention mechanism, NCI models (Wang et al. 2022) now use such a mechanism to infer item-item relationships based on user histories. Similarly, the SASRec architecture (Kang and McAuley 2018) proposes a self-attention based sequential model to capture long-term semantics.

Also, the rise of language models such as BERT (Devlin 2019) and GPT (Radford 2018) has spurred various language model studies with different objective functions for recommender systems. For example, BERT4Rec (Sun et al. 2019) employs bidirectional self-attention to learn sequence histories. According to a survey by Wu (2023), autoregressive models, which use decoder-only architectures, are better at capturing long-term dependencies, leading many studies to focus on these models. Other approaches, like IRGAN and DiffRec, explore the use of diffusion mechanisms and GANs in recommender systems (Wang et al. 2023). With the emergence of pre-trained large language models (PLMs), models like P5 (Geng et al. 2022) and M6 (Cui et al. 2022) have applied PLMs to multi-class recommender systems.

Recently, there have been a few efforts to apply generative retrieval techniques to recommendation tasks. For instance, the Transformer Index for Generative Recommenders (TIGER) leverages RQ-VAE to generate semantic IDs derived from text embedding vectors produced by Sentence-T5 (Ni et al. 2021) and utilizes T5 (Raffel et al. 2020) to identify patterns in user histories (Rajput et al. 2024). Recent research also adds contrastive learning to TIGER so it can understand not only semantic meanings but also relationships between items (Jin et al. 2024). Similarly, our research uses the RQ-VAE model to generate semantic IDs, replacing traditional item IDs and user IDs. To the best of our knowledge, we are the first to generate both user semantic IDs and item semantic IDs (e.g., government subsidy policy documents) for generative retrieval recommender systems.

Regarding the deployment of recommender systems, Jannach et al. (2022) argued two critical limitations in existing recommendation system research: an overemphasis on "one-shot" performance metrics, and the absence of practical evaluation methodologies such as A/B test, coupled with insufficient stakeholder engagement. This study addresses these limitations by expanding the evaluation framework beyond single-point metrics with recall@5, ndcg@5, recall@10, and ndcg@10. Furthermore, we employed the A/B test for practical validation and actively collaborated with industry stakeholders to fine-tune model size and hyper-parameters, thereby ensuring practical implication of our approach.

In addition, similar to the past research (Leony et al. 2013, Wanyi 2022), we used Amazon Web Service (AWS) rather than on-premises to deploy our AI-based recommender systems. As Wello is an emerging start-up with limited IT resources, we needed elastic architecture using Auto Scaling Group to minimize operational overhead and efficient resource utilization.

Research Dataset

Wello is a government tech firm that has gathered and promoted government subsidy information to users. Fig1 and Fig 2 are screenshots of Wello's sign-in page. In the sign-in process, Wello asks for and collects users' demographic information and their preferences in subsidy programs. For example, in Fig 1, the sign-in page collects users' demographic information including marriage status, income level, region (e.g., where users live), sex, education level, and job description, etc. In Fig 2, Wello asks users to report what types of government subsidy program they are interested in.

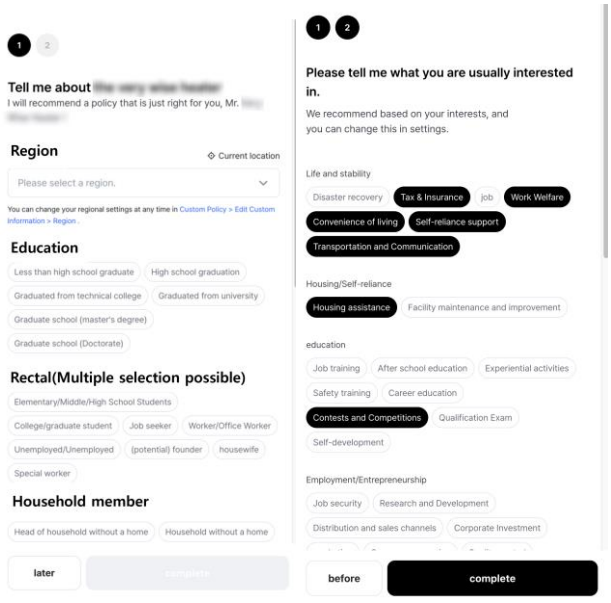

Figure 1: The first and second user sign-in page on the Wello platform

Thus, we use 185,610 user click-stream histories including users' demographic information, their preference histories, and their viewing histories with 240,000 subsidy documents, including their like and wish information. The user click-stream histories' maximum, minimum, and average are 460, 2, and 8.45, respectively.

Recommender System Architecture

Our generative recommender system architecture consists of two parts: quantizing embedding vectors into semantic IDs using our new RQ-VAE model and generating semantic IDs of potential subsidy program for the recommender system (Figure 2). First, we train our new RQ-VAE model to quantize embedding vectors into semantic IDs by choosing the indices of the closest codeword vector from codebooks. Specifically, government subsidy documents are embedded as training data by SentenceT5. Once training is done, the model generates a lookup table that maps the original document IDs and user IDs to the generated semantic IDs . Second, we train a transformer model on sequential recommendation tasks using the lookup table to map original document IDs and user IDs to corresponding semantic IDs. Finally, our generative recommender system outputs recommendations for potential subsidy programs that are relevant to a particular user.

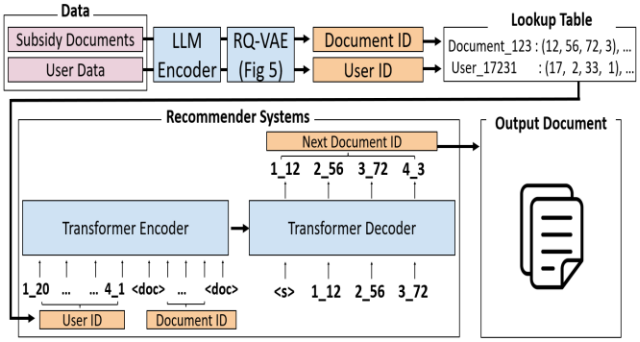

Figure 2: The process of generative recommender systems

New RQ-VAE Models

As mentioned above, we use the generative retrieval technique with a new RQ-VAE model for our generative recommender system. However, recent research has shown that the current RQ-VAE's first codebook and subsequent codebooks are imbalanced in capturing important features because the first codebook captures a majority of the semantic importance (Zheng et al. 2024). Since encoded vectors are subtracted from the closest codeword, their density will be extremely large at the last codebook step, which might lead to an imbalance between codebooks. Thus, we need to develop our new RQ-VAE model with multiple encoders and decoders between codebooks.

Figure 3 shows how our new RQ-VAE model works. The model consists of two VQ-VAEs (in practice RQ-VAE model consists of three VQ-VAEs). Unlike existing VQ-VAE models, each VQ-VAE in our new model gets a residual vector. Every input to a VQ-VAE layer is computed with the following formula: $x_i = \mathrm{x}_{i-1} - Decoder_{i-1}\left(Q\left(Encoder_{i-1}(x_{i-1})\right)\right)$. We set $x_0 \in \mathbb{R}^d$ as the embedding vector for either the user's information or the subsidy documents through SentenceT5, which could be replaced by pre-trained text encoders such as BERT. Q denotes a function that finds the closest codeword in a codebook, while *i* denotes the *i*th VQ-VAE layer. We can formulate this as $argmin\left||\boldsymbol{z_i} - \mathbf{e_k}|\right|$, where z_i is an encoding vector $\mathrm{z_i} = Encoder_i(x_i)$, while an e_k is a codeword vector from codebook $C_i \coloneqq \{e_k\}_{k=1}^{K}\, where\ i = 0, \dots, m-1$. Note that we define $||\cdot||$ as a Euclidean norm.

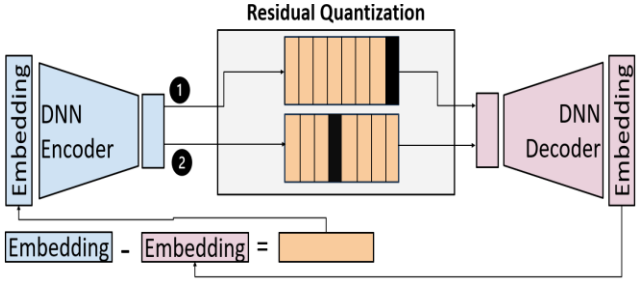

Figure 3: The architecture of our new RQ-VAE model

Here, we denote the number of layers, which is the number of total codebooks, as *m* and the cardinality of the codebook as *K*. For example, if we set *m* to 3 and *K* to 256, the semantic IDs of a subsidy document could be (234, 12, 129). As we use the residual vector as an input, this document should be more similar to (234, 12, 27) than (234, 129, 12). While existing RQ-VAEs use a single encoder-decoder, with multiple codebooks between encoder and decoder (Zeghidour et al. 2021, Adiban et al. 2022), we insert an encoder-decoder between codebooks because encoding residuals has benefit for capturing features.

After we get a quantized representation of e_i, which is the closest codeword vector in codebook C_i, through the function Q, e_i is used to reconstruct *x* through the decoder $\hat{x}_i = Decoder_i(e_i)$. Then, we compute the new RQ-VAE loss with reconstruction loss and commitment loss (or rqvae loss) (Van and Vinyals 2017, Zeghidour et al. 2021, Adiban et al. 2022). To prevent codebook collapse, we also initialize our codebook using k-means clustering-based initialization, as proposed in previous research (Rajput et al. 2024).

To summarize, we first use SentenceT5 to embed user and document information. Then, the first layer of our encoder-codebook-decoder quantizes (i.e., finds the closest codeword vector) the embedding vector using the same process as a VQ-VAE. After that we subtract the embedding vector from the decoding vector and move to the next layer's input. Lastly, our new RQ-VAE model repeats this process for the total number of codebooks.

Preprocessing for Recommender Systems

Once we finish training our model, we generate a lookup dictionary that converts user IDs and document IDs to semantic IDs. For example, $\{doc_1 : (24,18,128,1), \text{doc}_2 : (24,18,128,0), \ldots, user_3 : (129,42,255,0), \ldots\}$ where m=3 and K=256. Here, the last digit distinguishes items that have the same three-digit number. Through this lookup dictionary, we convert user IDs and document IDs in the user-click history to semantic IDs. For example, let $(\text{user}_{24}, \text{doc}_{12}, \text{doc}_{472})$ be user 24's history. We convert this sequence to $(s_{24,1}, s_{24,2}, s_{24,3}, s_{24,4}, <\text{doc}>, s_{12,1}, s_{12,2}, s_{12,3}, s_{12,4}, <doc>, s_{472,1}, s_{472,2}, s_{472,3}, s_{24,4})$, where $s_{id,m}$ means either $user_{id}$'s m_{th} digit or doc_{id}'s m_{th} digit in the corresponding semantic ID, while <doc> is a special token for separating semantic IDs and letting the model know semantic IDs will be followed. Unlike previous studies (Rajput et al. 2024), we convert not only document IDs but also user IDs into semantic IDs. We believe this will make our recommender system into a more personalized recommendation algorithm.

Recommender System Architecture

By using user click-stream histories, which are represented by semantic IDs generated through a lookup dictionary, we develop our AI-based recommender system, which we call the generative recommender system here. Following prior research (Tay et al. 2022, Sun et al. 2024, Rajput et al. 2024), we select the transformer architecture for our recommendation system. While the original transformer architecture is designed to predict the next word or token, our architecture is adapted to predict the next semantic ID, making it a sequence-to-sequence recommendation task. Specifically, the encoder in our generative recommender system transforms the input sequence into a contextualized vector using the attention mechanism. The decoder then predicts the next semantic ID based on this contextualized vector and the model's previous predictions. As a result, our generative recommender system can process both user information that is represented by user semantic IDs and user histories that are composed of document semantic IDs, and then it returns the output: a recommendation of subsidy programs that fit users.

Model Setting

We chose SentenceT5 to convert subsidy documents and user information into 768-dimension embedding vectors. Our new RQ-VAE model has 4 codebooks, and each codebook has 256 cardinalities. Each encoder layer has 768-dimension, and 512-dimension layer with 256 final output dimensions. Between these layers, our model includes a ReLU activation function. For our generative recommendation systems, we choose 128 embedding dimensions, 512 feed forward dimensions, 4 attention heads, and 3 layers for the encoder and decoder.

Regarding the model parameters, we follow the transformer architecture configuration described in Rajput et al. (2024), with a scaled-down version of the transformer architecture to meet Wello's requirement for a more cost-efficient model. We selected our scaled-down hyperparameters by following conventional transformer design principles - maintaining the feed-forward dimension (which is 512) at four times the embedding dimension (which is 128) and ensuring the embedding dimension (which is 128) remains divisible by the number of attention heads (which is 4) (Rajput et al., 2024).

Results

Results with Wello Dataset

We first trained our new RQ-VAE model on the Wello dataset with a 4e-1 learning rate, 4,096 batch size and Adagrad optimizer. Before we moved on to the generative recommender system training, we needed to see which cardinalities and what number of codebooks are the best option for Wello datasets. Thus, we tested eight versions of RQ-VAE, namely 32/6, 32/8, 64/6, 64/8, 64/10, 128/8, 128/10, and 256/4, where each format represents Cardinality/Number of Codebooks. We found that the RQ-VAE models with 6 codebooks with 32 cardinalities (i.e. 32/6) and 4 codebooks with 256 cardinalities (i.e. 32/6) showed the best average usage rate and lowest validation loss, respectively, among all options.

Then, for both the 32/6 and 256/4 models, we prepared a recommendation dataset using the same approach as in preparing for the recommender systems section. Specifically, we converted the user clickstream history to semantic ID sequences. For example, we converted user history $(\text{user}_{24}, \text{doc}_{12}, \text{doc}_{472})$ to the following semantic ID sequences using the 256/4 model: $(<s>, 1\_s_{24,1}, 2\_s_{24,2}, 3\_s_{24,3}, 4\_s_{24,4}, <\text{doc}>, 1\_s_{12,1}, 2\_s_{12,2}, 3\_s_{12,3}, 4\_s_{12,4}, <doc>, 1\_s_{472,1}, 2\_s_{472,2}, 3\_s_{472,3}, 4\_s_{24,4}, </s>)$, where "1_", "2_", "3_", and "4_" represent the digit location in the semantic ID, used to distinguish the same number in different digits, while $s_{24,1}$ refers to the first digit of the semantic ID of an item whose ID is 24. Thus, if document 24's semantic ID is (12, 23, 23, 4), it will convert to (1_12, 2_23, 3_23, 4_4). Note that if we use the 32/6 model, each semantic ID has 6 digits.

Next, we trained our generative recommender systems for 10 epochs with a learning rate of 1e-3, batch size 4096, LambdaLR scheduler, and AdamW optimizer (decay = 0.01). We used the same training method and the same recommender system architecture with different RQ-VAE models (256/4 and 32/6) to compare their recommendation

performance. In Table 1, we can see the 256/4 model always shows slightly better performance than the 32/6 model. Therefore, we decided to use the 256/4 model for our generative recommender system.

Additionally, we show how our model's performance varies depending on user semantic IDs. As we noted, this is the first study to use both subsidy documents and user information (e.g., demographic information, personal preference, and "like" or "wish" logs on subsidy programs) in the generative retrieval recommender system literature. Table 1 shows that removing user IDs led to an average performance decline of 4.5% in the recommender system algorithm. These results suggest that user IDs contribute to improving the overall performance of our generative recommender systems.

Methods	Wello Dataset			
	Recall @5	NDCG @5	Recall @10	NDCG @10
32/6	0.3989	0.2911	0.4844	0.3190
256/4 (base)	**0.4084**	**0.2964**	**0.5017**	**0.3264**
w/o user ID	0.3863 (-5.4%)	0.2868 (-3.2%)	0.4730 (-5.7%)	0.3144 (-3.7%)

Table 1. Results of our model on the Wello dataset

Models	Sports and Outdoors			
	Recall @5	NDCG @5	Recall @10	NDCG @10
P5	0.0061	0.0041	0.0095	0.0052
GRU4Rec	0.0129	0.0086	0.0204	0.0110
BERT4Rec	0.0115	0.0075	0.0190	0.0099
SASRec	0.0233	0.0154	0.0350	0.0192
TIGER	0.0264	0.0181	**0.0400**	0.0225
Our Gen-Rec	**0.0267**	**0.0182**	0.0395	**0.0227**

Table 2. Performances of our generative recommendation system on Amazon (Sports and Outdoors) dataset

Results with Amazon Dataset

To compare our model's performance with that of existing generative recommender systems, Table 2 shows the performance of our recommender system with Amazon's Sports and Outdoors dataset. Most of the training settings are the same as in previous research (Rajput et al. 2024, Jin et al. 2024). Furthermore, we use the leave-one-out method, as also used in previous research (Sun et al 2019, Rajput et al. 2024). Specifically, the leave-one-out method is a sequence-to-sequence training method with prediction for the very last item (for us, the 4-digit semantic ID generated by 256/4 model) in the test set, the second last item for the validation set, and the third last item for the training set. We find that our generative recommender system's performance is similar to other models in previous studies (Rajput et al. 2024).

Deployment

Based on the successful performance, we began to deploy our new AI-based recommender system on the Wello platform in the second quarter of 2024. The Wello technical team is currently responsible for maintaining the generative recommender system and continuously updating both the model and the database to ensure optimal performance. This involves accommodating new government subsidy programs, as well as adapting the changes in users' preferred subsidy programs (as shown by them clicking "like" or "wish" on given subsidy programs as well as their viewing histories). By incorporating updated user clickstream histories, our model is retrained every month to keep the recommendations accurate and relevant.

Figure 4 illustrates the details of the deployment process of our generative recommender system. It operates through a workflow that processes policy document data and user data to generate personalized recommendations. The system begins by collecting policy document data that the user has previously interacted with, and user data stored in Aurora DB for each user. This data undergoes preprocessing and filtering to create structured information that can be effectively utilized by the recommendation engine. The recommendation engine then integrates these two data streams - the user preferences data and user behavioral data – inferencing candidate policy documents and applying filtering algorithms to match user attributes with policy attributes. Through this process, the system implements personalization algorithms based on user patterns. These personalized policy recommendations are then formatted and presented to users through the web interface.

To show how our generative recommender system works, we randomly pick one piece of sample in the test dataset. In Figure 5, there is a young user seeking a job in Seoul, South Korea. Her interests include subsidy programs related to career development education, support for the cost of living, and support for public transportation costs. Based on this information, our recommender system suggests the following subsidy programs on the platform: a free education program at the government platform for career development, K-pass

(e.g., subsidies for public transportation for younger generations), and government loans to cover the cost of living for younger generations (i.e., under age 24). If she clicks one of these recommended subsidy programs, she can see all the program details, including how to apply, what documents she needs, the due date, and benefits the program offers. We therefore believe our generative recommender system integrates users' preferences and viewing histories to recommend the most suitable government subsidies for them. With our generative recommender system, users at Wello platform no longer need to spend time searching for relevant subsidy programs, or risk missing out on essential subsidy programs. We therefore believe our generative recommender system integrates users' preferences and viewing histories to recommend the most suitable government subsidies for them. With our generative recommender system, users at Wello platform no longer need to spend time searching for relevant subsidy programs, or risk missing out on essential subsidy programs.

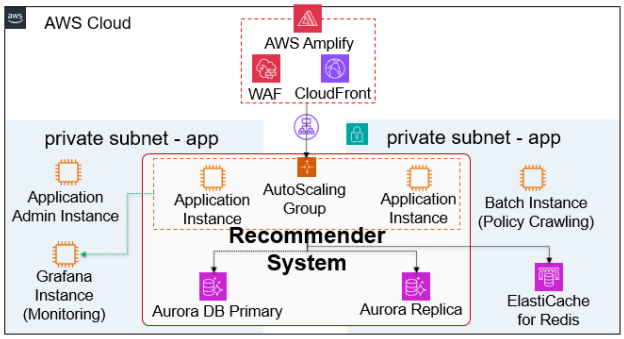

Figure 4: Deployment process in Amazon Web Service

Business Implications

To observe our recommendation effect, Wello conducted the A/B tests in the third quarter of 2024 and the fourth quarter of 2024 and observed a significant improvement in some business metrics: Click-Through Rate (CTR), average time spent per user, and survey results on satisfaction level with our new generative recommender systems. Although Wello began deploying our new AI-based recommender system on the platform in March 2024, additional time was needed to roll it out fully to all customers. Between March 2024 and August 2024, we have tested and monitored the performance of the recommender system algorithm, keeping update it continuously. By the end of the third quarter of 2024, Wello successfully opened the new recommender system for all customers and updated the UX/UI accordingly. So, to evaluate our new recommender system's effect, we can randomly select 1,000 customers who have experienced Wello's previous key-word-based recommender systems only (not our new AI based recommender system) during the third quarter of 2024 while we also randomly select 1,000 customers who have experienced our new recommender systems only during the fourth quarter of 2024. Figure 6 shows the improvements in our A/B Tests from a baseline with Wello's past recommender system (i.e., based on some keywords mentioned above) to our new AI-based recommender system. During this period, the results show that Wello has experienced a 68% Click-Through-Ratio (CTR) improvement from 41.4% to 69.6%. Additionally, average time spent per user increased by 14.3%, from 420 seconds to 480 seconds in the same period. Last, the survey results showed a 19.8% improvement in service satisfaction thanks to our generative recommender system.

Government Implications

In this section, we discuss the benefits to the Korean government of our generative recommender system. First, we noted a 36% increase in the number of applications to government subsidy programs on the platform after implementing our generative recommender systems. This suggests a potential decrease of 36% in unspent funds, amounting to $7.8 million[2] that would otherwise remain unused due to insufficient applications. Second, we find that the total advertising cost of the Korean government is around $950 million. Thus, we can estimate the potential advertising saving cost as follows: 36% increase in the number of subsidy program applications X Wello's current market share (8%) X total advertising cost for promoting subsidy programs at South Korea ($950 million) = $27.4 million. Also, we confirm that one Korean government officer reported that our generative recommender system indeed saved around $27 million in advertising expenses for subsidy policy programs.

Conclusion

In this work, we present a novel generative recommender system for government subsidy programs by using a retrieval generative model. We develop a new RQ-VAE model to capture deep and abstract features, generating semantic IDs that represent the content of subsidy programs. Our generative recommender system operates by using user

[2] The total amount of unused amounts of Korean government's budget due to lack of application ($2.7 billion) X 36% increase in the number of users who applied for the government program X Wello's current market share (8%) = $7.8 million. Also, we get Wello's current market share as follow: the average number of Wello's active users in 2024 (around 3 million) % the number of adult populations in South Korea (around 4,000 million) = 8%. Please refer to the report below. http://www.narasallim.net/report/659

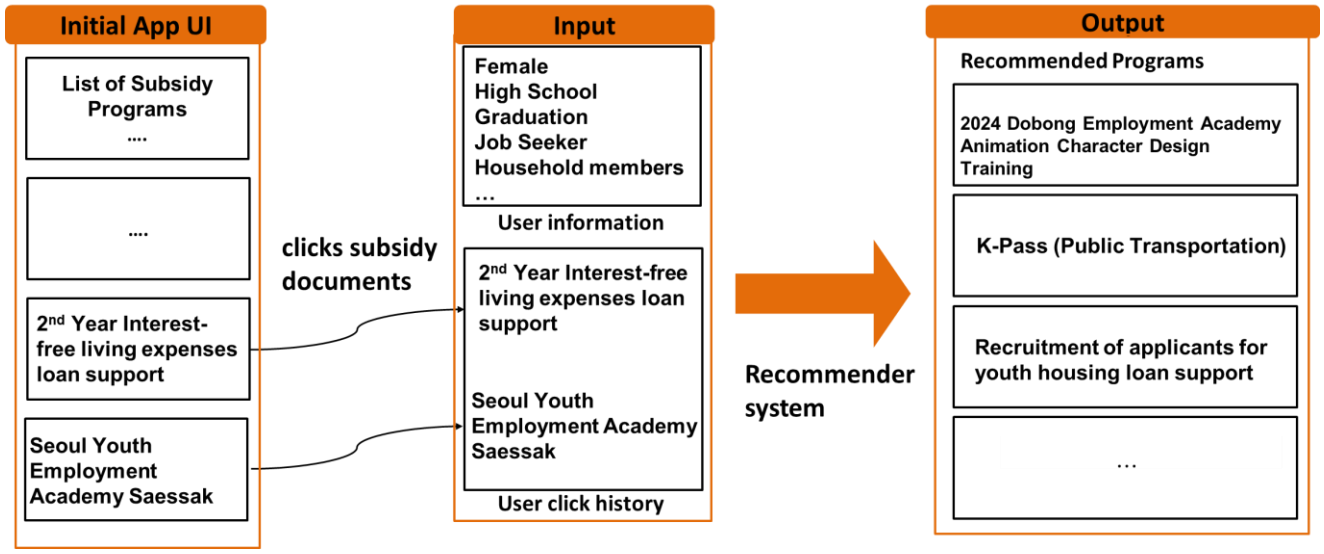

Figure 5: Examples of the deployment of our generative recommender system

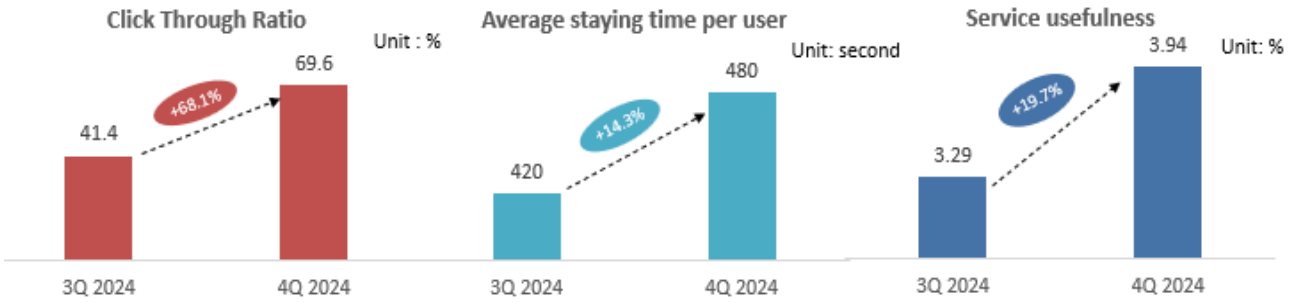

Figure 6: Results of CTR, average engagement time per user, and survey results

semantic IDs (i.e., users' demographic information and their preferences for subsidy programs) and document semantic IDs (i.e., subsidy program documents). Our generative recommender system performs well compared to those from other studies. Additionally, we reveal that our generative recommender system improves some business metrics, such as click-through rate (CTR), average user engagement time, and user survey results. These findings suggest that our new generative recommender system could potentially help the Korean government by reducing the advertising subsidy policy cost and reducing the amount of budgets being unspent due to a lack of applications.

This study has a few limitations. Some questions remain regarding our generative recommender system's performance across other datasets and model sizes. Incorporating additional techniques, such as contrastive learning (Jin et al. 2024), may further improve the performance of our model. Future studies need to make degradation metrics rather than the calendar basis to retrain our model and maintain its performance constantly. Also, it would be great if future studies would examine our model performance with more relevant data sets instead of Amazon dataset. Last, we haven't yet compared our model performance with traditional recommender system algorithms (e.g., content-based filtering or collaborative filtering). We hope that future studies will report a more detailed performance of our AI-based recommender system compared with the baseline model.

Acknowledgments

This work was supported by Wello Inc. The Wello and research team specially thank to The Presidential Committee on the Digital Platform Government in South Korea for launching the Wello's services.

References

Adiban, M.; Siniscalchi, M.; Stefanov, K.; and Salvi, G. 2022. Hierarchical Residual Learning Based Vector Quantized Variational

Autoencoder for Image Reconstruction and Generation. In Proceedings of the 33rd British Machine Vision Conference. London, UK: BMVA Press.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(8): 1798 –1828.

Cui, Z.; Ma, J.; Zhou, C.; Zhou, J.; and Yang, H. 2022. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. arXiv preprint. arXiv:2205.08084 [cs.IR]. Ithaca, NY: Cornell University Library.

Devlin, J. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint. arXiv:1810.04805 [cs.CL]. Ithaca, NY: Cornell University Library.

Embrain. 2023. 2023 Focus Media Advertising Performance Analysis Report. Internal report. Seoul: Embrain.

Geng, S.; Liu, S.; Fu, Z.; Ge, Y.; and Zhang, Y. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In Proceedings of the 16th ACM Conference on Recommender Systems, 299-315. New York: Association for Computing Machinery.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2015. Session-Based Recommendations with Recurrent Neural Networks. arXiv preprint. arXiv:1511.06939 [cs.LG]. Ithaca, NY: Cornell University Library.

Hou, Y.; He, Z.; McAuley, J.; and Zhao, W. X. 2022. Learning Vector-Quantized Item Representation for Transferable Sequential Recommenders. arXiv preprint. arXiv:2210.12316 [cs.IR]. Ithaca, NY: Cornell University Library.

Jannach, D.; Pu, P.; Ricci, F.; and Zanker, M. 2022. Recommender systems: Trends and frontiers. Ai Magazine, 43(2), 145-150.

Jin, M.; Qiu, Z.; Zhu, J.; Dong, Z.; and Li, X. 2024. Contrastive Quantization Based Semantic Code for Generative Recommendation. arXiv preprint. arXiv:2404.14774 [cs.IR]. Ithaca, NY: Cornell University Library.

Kang, W.-C.; and McAuley, J. 2018. Self-Attentive Sequential Recommendation. In Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM), 197–206. Los Alamitos, CA: IEEE Computer Society.

Lee, H.; Kim, J.; Chang, H.; Oh, H.; Yang, S.; Lu, Y.; and Seo, M. 2022. Contextualized generative retrieval.

Leony, D.; Gélvez, H. A. P.; Merino, P. J. M.; Pardo, A.; and Kloos, C. D. 2013. A Generic Architecture for Emotion-based Recommender Systems in Cloud Learning Environments. *J. Univers. Comput. Sci.*, *19*(14), 2075-2092.

Moreira, G. S. P.; Rabhi, S.; Lee, J. M.; Ak, R.; and Oldridge, E. 2021. Transformers4Rec: Bridging the Gap between NLP and Sequential/Session-Based Recommendation. In Proceedings of the Fifteenth ACM Conference on Recommender Systems, 143-153. New York: Association for Computing Machinery.

Ni, J.; Abrego, G. H.; Constant, N.; Ma, J.; Hall, K. B.; Cer, D.; and Yang, Y. 2021. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. *arXiv preprint arXiv:2108.08877*.

Pathak, B.; Garfinkel, R.; Gopal, R. D.; Venkatesan, R.; and Yin, F. 2010. Empirical Analysis of the Impact of Recommender Systems on Sales. Journal of Management Information Systems 27(2): 159-188.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-Training. Technical report. San Francisco, CA: OpenAI.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; ... and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, *21*(140), 1-67.

Rajput, S.; Mehta, N.; Singh, A.; Keshavan, R. H.; Vu, T.; Heldt, L.; Hong, L.; Tay, Y.; Tran, V.; Samost, J.; et al. 2024. Recommender Systems with Generative Retrieval. In Advances in Neural Information Processing Systems 36. Cambridge, MA: MIT Press.

Singh, A.; Vu, T.; Keshavan, R.; Mehta, N.; Yi, X.; Hong, L.; Heldt, L.; Wei, L.; Chi, E.; and Sathiamoorthy, M. 2023. Better Generalization with Semantic IDs: A Case Study in Ranking for Recommendations. arXiv preprint. arXiv:2306.08121 [cs.IR]. Ithaca, NY: Cornell University Library.

Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; and Jiang, P. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 1441-1450. New York: Association for Computing Machinery.

Sun, W.; Yan, L.; Chen, Z.; Wang, S.; Zhu, H.; Ren, P.; ... and Ren, Z. 2024. Learning to Tokenize for Generative Retrieval. In Advances in Neural Information Processing Systems 36. Cambridge, MA: MIT Press.

Tay, Y.; Tran, V.; Dehghani, M.; Ni, J.; Bahri, D.; Mehta, H.; ... and Metzler, D. 2022. Transformer Memory as a Differentiable Search Index. In Advances in Neural Information Processing Systems 35, 21831-21843. Red Hook, NY: Curran Associates, Inc.

Van Den Oord, A.; and Vinyals, O. 2017. Neural Discrete Representation Learning. In Advances in Neural Information Processing Systems 30, 6306-6315. Red Hook, NY: Curran Associates, Inc.

Wang, Y.; Hou, Y.; Wang, H.; Miao, Z.; Wu, S.; Chen, Q.; ... and Yang, M. 2022. A Neural Corpus Indexer for Document Retrieval. In Advances in Neural Information Processing Systems 35, 25600-25614. Red Hook, NY: Curran Associates, Inc.

Wang, W.; Xu, Y.; Feng, F.; Lin, X.; He, X.; and Chua, T. S. 2023. Diffusion Recommender Model. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 832–841. New York: Association for Computing Machinery.

Wanyi, P. 2022. Analysis of Personalized Recommendation System on Cloud Platform. *Journal of Innovation and Social Science Research*.

Wu, L.; Zheng, Z.; Qiu, Z.; Wang, H.; Gu, H.; Shen, T.; ... and Chen, E. 2023. A Survey on Large Language Models for Recommendation. arXiv preprint. arXiv:2305.19860 [cs.IR]. Ithaca, NY: Cornell University Library.

Zeghidour, N.; Luebs, A.; Omran, A.; Skoglund, J.; and Tagliasacchi, M. 2021. Soundstream: An end-to-end neural audio codec. arXiv preprint. arXiv:2107.03312 [eess.AS]. Ithaca, NY: Cornell University Library.

Zheng, C.; Vuong, T. L.; Cai, J.; and Phung, D. 2022. MoVQ: Modulating Quantized Vectors for High-Fidelity Image Generation. In Advances in Neural Information Processing Systems 35, 23412-23425. Red Hook, NY: Curran Associates, Inc.