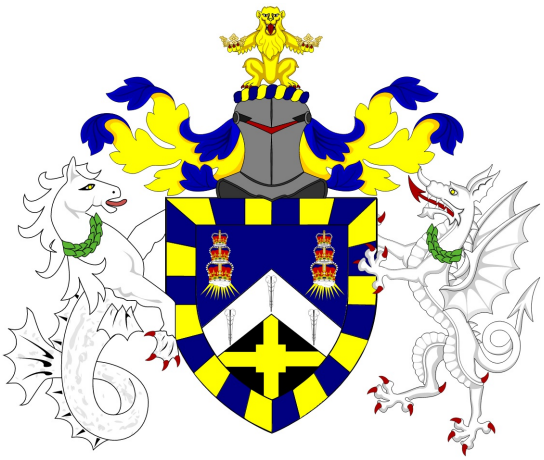
Data Analytics MSc Dissertation MTH797P, 2019/09

Yelp Restaurant Recommendations using Matrix Factorisation and Variational Autoencoder

Meilin Lai, ID 036782976

Supervisor: Dr. Martin Benning



A thesis presented for the degree of

Master of Science in *Data Analytics*

School of Mathematical Sciences
Queen Mary University of London

Declaration of original work

This declaration is made on September 11, 2020.

Student's Declaration: I Meilin Lai hereby declare that the work in this thesis is my original work. I have not copied from any other students' work, work of mine submitted elsewhere, or from any other sources except where due reference or acknowledgement is made explicitly in the text, nor has any part been written for me by another person.

Referenced text has been flagged by:

1. Using italic fonts, **and**
2. using quotation marks "...", **and**
3. explicitly mentioning the source in the text.

Acknowledgements

Special thanks to my dermatologist who magically fixed all the acnes I got from completing this dissertation. Also thanks to my friend April for buying me the tomato egg soup that somehow cheered me up when I was struggling with debugging. And thanks to Anna who for some reason, still wants to be friend with me after hearing me talking about my dissertation over and over again.

Abstract

Accompanied by the proliferation of the online economy, recommendation systems have become an integral part of internet services. The current study focuses on generating Yelp restaurant recommendations using two collaborative filtering based models: Matrix Factorisation (MF) and Variational Autoencoder (VAE). Both models are trained and evaluated using implicit data which takes the form of binarised explicit ratings. Despite its increasing popularity, implicit data has been associated with some longstanding challenges that are primarily concerned with the ambiguous nature of the zero entries. It has been repeatedly pointed out that although the positive feedback is observed with confidence, the negative feedback is not, since the zero values could either be interpreted as a sign of disinterest (negative feedback) or items that users are genuinely not aware of (missing feedback). The current thesis argues that although this is the case for a certain type of implicit data such as count information, the same does not apply to binarised explicit ratings, as the negative feedback is also observed with confidence. This, however, does not imply that the zero entries are not without ambiguities, as they could either represent unreported feedback or information that is truly missing as typically assumed. A sampling scheme is, therefore, proposed to address such an issue by selecting a subset of entries to be the unreported feedback. Empirically, results from the current study provides the experimental evidence for the superior performance of VAE over MF model. On top of this, the proposed sampling scheme is also shown to be effective in improving the model's ability in filtering out irrelevant items. Furthermore, we also conduct a set of experiments examining the bias-correcting effect of a newly proposed reweighing scheme and have demonstrated favourable results. Finally, extended experiments are carried out to

investigate the effect of incorporating one-hot encoded item category information through pretraining. The results do not provide any empirical support for the value of such a method in improving recommendation quality.

Contents

Chapter 1

Introduction

As the web grows in size, people are presented with an increasingly large amount of online information, which could be overwhelming when not utilised efficiently. More effective use of online content could be promoted by a well-designed recommendation systems (RS) that act as an information filtering mechanism that only present users with items that they are likely to enjoy. There are in general, four types of input that a recommendation system is built upon: user-item interaction histories, user preferences, item features, and other environmental factors such as social relationships, time etc. Different combinations of input data often imply different types of model design [7]. The three most commonly adopted choices are content-based RS, collaborative filtering (CF) and hybrid models. Content-based RS utilises information about user preferences or item features, whereas CF only focuses on historical interaction records. Hybrid models seek to learn a richer representation by integrating two or more types of input data. The current thesis limits its scope to the last two types of model due to their superior performance and increasing popularity [21].

1.1 Recommendation Tasks for Collaborative Filtering

In the real-world setting, a user typically interacts with only a small fraction of items, leaving the remaining ones as the recommendation candidates, from which an RS will identify a subset of items that the user is likely to enjoy. This could be achieved through completing one of the two types of recommendation tasks: ranking prediction and feedback prediction. The former involves producing a ranked list of items and selecting the top $\mathcal{N}$ items as the ones that are most likely to be preferred by the user (also known as Top-$\mathcal{N}$ recommendation). The latter focuses on predicting the feedback information of the candidate items and then selecting recommendations based on the predicted feedback scores. This task has often been reframed as a matrix completion problem where user u's preference for item i is represented as the entry r_{ui} in a user-item interaction matrix R. Since preference information is not available for most of the items, the matrix completion task involves filling in the missing preferences while ensuring the observed entries are matched. Both tasks are implemented in this project to gain a more comprehensive understanding of the model performance. A stronger emphasis is placed on the latter as it forms the basis of our model comparison.

1.2 Treatments for Binarised Ratings

Users' feedback information usually takes two forms: explicit and implicit. Explicit feedback refers to the raw ratings score often measured on a five-point Likert scale, whereas, implicit feedback typically refers to clicks and logs data [8]. Although it has recently become a common practice to also treat binarised explicit ratings as implicit data, the current thesis argues that the common treatments for the traditional implicit data might not be suitable for binarised explicit data and henceforth, introduces a new sampling scheme catered for binarised ratings. The sampling scheme is based on the observation that not all missing entries are uninformative in the absolute sense, as some could be items that the user has previously interacted with but of which its feedback information was not reported

to the system. Hence, to account for such a possibility, a subset of missing entries is sampled as unreported feedback with its value computed through a weighted average between the user's average rating and item average ratings. The number of entries sampled is dependent on the user's activity level. Although theoretically, such a sampling scheme could improve model performance by reducing the sparsity of the interaction matrix, it is yet to be validated empirically. Therefore, the current study investigates such a proposition by comparing model performance with and without our new sampling scheme.

1.3 Simplifying the Feedback Prediction as a Classification Task

Conventionally, the models designed for feedback prediction tasks generally use explicit ratings and focus on minimising the squared Euclidean distance between the reconstructed entries and their corresponding ground-truth values. This often leads to the problem where the reconstructed entries exceed the scale on which the observed entries are measured. This problem could be overcome by transforming the prediction task into a classification task by focusing on binarised ratings instead. This not only simplifies our prediction task but also makes the prediction results more consistent with the overarching aim of a recommendation system, which primarily concerns the decision whether or not an item should be recommended, rather than knowing the exact rating of an item. Therefore, each entry in the input data is assumed to follow a Bernoulli distribution for all models implemented. More detailed explanations of how such an assumption is incorporated into each model is presented in the model description section.

1.4 Related Studies

Studies that adopted CF-based methods predominantly incorporated the latent factor model as their core working mechanism, which is predicated on the assumption that the observed feedback could be explained by some underlying information

about customer types and item attributes, which are referred as latent variables [3]. These latent variables are often retrieved by factorising the sparse user-item interaction matrix into two low rank matrices. The product of the two matrices then forms the reconstructed interaction matrix with filled missing entries. This process forms the basis of Matrix Factorisation (MF) models, which could be optimised using algorithms such as alternating minimisation [15] and alternating least square [17]. It is worth mentioning that since the reconstructed entry is computed as the linear combinations of latent item features and latent user features, the core mechanism of MF models is inherently linear, and is often argued to be insufficient to capture the complex user-item interactions. This motivates the development of a deep generalisation of the conventional MF: autoencoder (AE). Its most basic version consists of three layers: input layer, hidden layer and output layer. The first two layers form the encoder which projects the partially observed user-item interaction matrix $R \in \mathbb{R}^{m \times n}$ into a K-dimensional latent space through the function $h = f_{s_1}(RW_1 + b_1)$, where $W_1 \in \mathbb{R}^{n \times K}$ and $b_1 \in \mathbb{R}^K$ are the weight and bias parameters and f_{s_1} is the activation function that captures the non-linearities in user-item interactions. The last layer forms the decoder network which projects the latent variables back to the output space to generate the reconstructed matrix R^* through: $R^* = f_{s_2}(hW_2 + b_2)$, where $W_2 \in \mathbb{R}^{K \times n}, b_2 \in \mathbb{R}^n$. The network could be made more complicated by adjusting the number of layers and the number of neurons in each layer.

This seemingly simple model constitutes the basis of Auto-rec [18], which is one of the first few attempts to build an autoencoder-based RS. Recent developments in the field have been primarily concerned with incorporating variants of autoencoders to improve its generalisation power [19]. One branch of studies extends Auto-rec to make the learned representations more robust to noises by leveraging denoising techniques that corrupt inputs using masking noise, Gaussian noise and salt-and-pepper noise [20][11]. Another branch of studies attempts to reformulate the problem in a Bayesian generative framework and adopts Variational Autoencoder (VAE) as their basis model. VAE differs from the plain and denoising AE in that it seeks to learn a probability distribution of the latent variable instead of its point estimate [22]. The probability distribution often takes the form of

a normal distribution where its variance is frequently interpreted as the added stochastic noises in the latent space. Since the stochasticity is now added in the latent space, instead of the observation space as the case for denoising AE. This eliminates the need to find the best input corruption scheme for a given dataset, which is another hyperparameter that requires tuning. For such a reason, VAE is chosen to be the non-linear counterpart of the aforementioned MF model. Although it is often argued that the ability to capture the non-linearities in user-item interactions allow the autoencoder-based recommendation systems to outperform the linear methods [4], previous studies have rarely provided any experimental evidence supporting such a claim. The current study contributes to the field by empirically examining this claim through comparing the performance of the linear baseline (conventional MF optimised using alternating minimisation algorithm) with VAE on the feedback prediction task. To generate more comparable results with the current literature, the VAE model is also modified to output a probability distribution over items for the ranking prediction task.

1.5 Incorporating Side Information

So far, the discussion has mainly revolved around CF-based models, the current section extends the discussion to include hybrid models, where side information such as item attributes or users' profile are utilised along with the historical interaction records. The ways in which side information is integrated vary depending on the type of basis model chosen (linear vs non-linear) and the type of side information considered. Some have used variants of AE to learn the latent features of the user and item side information. The learned features are then concatenated with their corresponding low dimensional matrices decomposed from the interaction matrix through probabilistic MF (a Bayesian formulation of the conventional MF) [1][11]. Others have used VAE to learn the latent features from both the interaction matrix and auxiliary information, but differ in how the learned features are integrated. In [4], the authors first use VAE to learn the movie embeddings from plot descriptions and modifies the interaction matrix with learned movie embeddings. A separate VAE was then used to learn the latent representation of

the modified interaction matrix. Unlike Gupta, Raghuprasad and Kumar[4], Lee et al.[10] and Chen et al.[1] use a single VAE to learn the latent features of both interaction and side information. However, in [1], two types of latent features were encoded in the same latent variable, whereas Lee et al.[10] uses two separate latent variables instead. Although all of the aforementioned studies have shown an improved model performance with the addition of side information, the result might differ depending on the dataset and/or type of auxiliary information considered. Following the method introduced in [1], the current study extends this line of research by exploring the effect of incorporating item category information on the Yelp dataset.

To summarise, the main contributions of the current thesis are listed as follows.

- Introduces a new sampling scheme for binarised explicit data to account for the unreported feedback.
- Simplifies the feedback prediction as a classification problem without deviating from the overarching objective of the recommendation task.
- Empirically examines the claim for the superior performance of the auto-encoder based recommendation systems over its linear counterpart.
- Explores the influence of integrating item category information into VAE models on the Yelp dataset.

Chapter 2

Preliminaries

2.1 Notations

Two types of user feedback have so far been discussed in the current thesis. For explicit feedback, the interaction matrix R contains values that are measured on a 5-Point Likert Scale, indicating the magnitude of preferences. Whereas, for implicit feedback, R only consists of zeros and ones, indicating binary preferences. A full list of notations used in this thesis is presented in Table 2.1.

2.2 Generative Latent Variable Model

Similar to MF, the Generative Latent Variable Model also uses latent features as part of their core modelling mechanisms. The main difference lies in that the latter reformulates the recommendation problem in a Bayesian framework by approaching it as a generation task. That is, it tries to ensure that for each user, there exists at least some realisations of the latent variable z, from which we could generate a rating matrix that is sufficiently similar to the input matrix on the observed entries. One simple solution for choosing z is to directly sample from its prior distribution $p(z)$. However, due to the high-dimensional nature of the sample space $\mathcal{Z}$, the generated ratings are unlikely to be similar to the input ratings for most of the z sampled. An alternative approach is to narrow down our sample

space by only sampling those z that are likely to have produced the observed ratings $P_\Omega(R)$. However, this requires us to compute the posterior distribution $p(z|P_\Omega(R))$, which is often intractable since the normalisation term $p(P_\Omega(R))$ is not available in closed form. In the following section, x is used in place of $P_\Omega(R)$ to avoid notational clutter.

2.3 Variational Inference

Variational inference contributes to the above problem by introducing a family of distributions $\mathcal{Q}$, where each $q(z) \in \mathcal{Q}$ is treated as a candidate approximation to the exact posterior. It then converts the inference problem into an optimisation problem by finding a member of $\mathcal{Q}$ that best approximates the true posterior. The performance of such approximations is measured with the Kullback–Leibler (KL) divergence, which by definition, is computed as follows.

$$KL[q(z)||p(z|x)] = E_{q(z)}[\log(\frac{q(z)}{p(z|x)})] \tag{2.1}$$

This leads to the following optimisation objective:

$$q^*(z) = \arg\min_{q(z)\in\mathcal{Q}} KL[q(z)||p(z|x)] \tag{2.2}$$

Expanding the KL divergence, and applying the Bayes rule, the dependencies on $p(x)$ is revealed.

$$KL[q(z)||p(z|x)] = E_{q(z)}[\log\, q(z)] - E_{q(z)}[\log\, p(x|z)] - E_{q(z)}[\log\, p(z)] + \log\, p(x) \tag{2.3}$$

Since $log\, p(x)$ does not depend on $q(z)$, the optimisation objective could be rewritten as:

$$\arg\min_{q(z)\in\mathcal{Q}} -E_{q(z)}[\log\, p(x|z)] + KL\,[q(z)\|p(z)] \tag{2.4}$$

The negative of equation (2.4) is often referred as the Evidence Lower Bound (ELBO) [3], minimising (2.4) is equivalent to maximising ELBO. The name comes from the fact that ELBO always serves as the lower bound of the $\log p(x)$. This

could be shown by substituting ELBO into (2.3):

$$\begin{aligned} \log\, p(x) &= KL[q(z)||p(z|x)] + \underbrace{E_{q(z)}[\log\, p(x|z)] - KL\left[q(z)\|p(z)\right]}_{\text{ELBO}} \\ &\geq \underbrace{E_{q(z)}[\log\, p(x|z)] - KL\left[q(z)\|p(z)\right]}_{\text{ELBO}} \end{aligned} \tag{2.5}$$

given that KL divergence is always non-negative. Equation (2.5) reveals the cornerstone of variational inference. It shows that by maximising ELBO, we have concurrently optimised two metrics of interest:

1. It minimises $KL[q(z)||p(z|x)]$, leading to a better approximation to the exact posterior

2. It maximises the lower bound of $\log p(x)$, which consequently improves the generative power of the model

Having introduced the main idea behind variational inference, the model description section explains how such an inference problem could be solved at scaled using Variational Auto-encoder.

Table 2.1: Notations used in this thesis.

Notations	Description
m	Number of users
u	A specific element in the user set
n	Number of items
i	A specific element in the item set
K	Dimensions of the latent variable
Ω	Set of indices of the observed entries
g	Batch size
v	Number of observed entries in R
$\mathcal{N}$	Number of recommended items in a ranked list
α	The correction coefficient
β	Coefficient for the KL divergence in the VAE loss function
λ	L2 regularisation coefficient in the MF loss function
η	Coefficient that controls the relative weighting of item and user rating mean
τ	learning rate
ρ	drop out rate
κ_u	Rating count for user u
μ_κ	Mean rating count averaged over all users
r_u^μ	Mean rating for user u (averaged over all items)
r_i^μ	Mean rating for item i (averaged over all users)
B_u	The set of positively rated items for user u.
f_{ui}	Output of the generation network for user u, item i
R	$\in \mathbb{R}^{m\times n}$ user-item interaction matrix
$R*$	$\in \mathbb{R}^{m\times n}$ reconstructed R with missing entries filled
P	$\in \mathbb{R}^{m\times k}$ matrix of latent user representation
Q	$\in \mathbb{R}^{k\times n}$ matrix of latent item representation
Y	$\in \mathbb{R}^{n\times d}$ matrix of side information
J	$\in \mathbb{R}^{m\times n}$ matrix of all ones
μ	$\in \mathbb{R}^{g\times k}$ the mean of variational distribution
σ^2	$\in \mathbb{R}^{g\times k}$ the variane of variational distribution
ϵ	$\in \mathbb{R}^{g\times k}$ dummy variable introduced for reparameterisation trick
z	$\in \mathbb{R}^{k}$ the latent variable
x_u	the set of known entries for user u
$P_\Omega(R)$	: $\mathbb{R}^{m\times n} \mapsto \mathbb{R}^v$ projection function that selects the known entries in R (often represented as x to avoid notational clutter)
$f_\phi(\cdot)$	The inference network
$f_\theta(\cdot)$	The generation network
$\mu(\cdot)$	The function that returns the mean of the variational distribution
$\sigma^2(\cdot)$	The function that returns the variance of the variational distribution
$\varsigma(\cdot)$	The sigmoid activation

Chapter 3

Model Descriptions

3.1 Matrix Factorisation (MF)

To formally formulate what has been introduced in the first section, the MF model attempts to solve the recommendation task by factorising R into two low rank matrices P and Q, such that the matrix product PQ gives rises the predicted feedback matrix R^* with the observed entries being correctly reconstructed, i.e.

$$R^* = PQ \quad \text{s.t. } P_\Omega(R) = P_\Omega(R^*) \tag{3.1}$$

Given the assumption that each entry in the binarised R follows a Bernoulli distribution, minimising the reconstruction errors is equivalent to minimising the negative logistic log-likelihood, which is shown as follows:

$$\begin{aligned}\mathcal{L}^{MF}(P,Q) = -\sum_{i=1}^{r}\Bigg(P_\Omega\Big(R \odot \log \varsigma(PQ) + (J-R)\odot \\ \log\left(J - \varsigma(PQ)\right)\Big)\Bigg)_i + \frac{\lambda}{2}(\|P\|^2 + \|Q\|^2)\end{aligned} \tag{3.2}$$

where $\varsigma(\cdot)$ and J respectively represents the sigmoid transformation and a $m \times n$ matrix with all ones. The l2 regularisation term was added to prevent over-fitting, with its magnitude being controlled by λ.

3.2 Variational Auto-encoder(VAE)

The current section is divided into four subsections. The first one discusses the choice of the variational distribution and output distribution in relation to the VAE architecture. Then the second subsection elaborates on the reparameterisation trick required for the model to be effectively trained. The third subsection offers an intuitive interpretation of the objective function and introduces the β coefficient. The last subsection explains how the side information is integrated into the current VAE model.

3.2.1 Scaling Variational Inference with VAE

Recall that in the preliminary section, we have derived the following objective function.

$$\mathcal{L}^{VAE} = \sum_{u=1}^{m} \Big(- E_{q(z_u)}[\log\, p(x_u|z_u)] + KL[q(z_u)||p(z_u)] \Big) \tag{3.3}$$

To compute $\mathcal{L}^{VAE}$, one needs to first be able to represent $p(z_u)$, $q(z_u)$ and $p(x_u|z_u)$. Common practice in mean-field variational inference typically sets $p(z_u)$ to be a standard Gaussian and $q(z_u)$ to be a fully factorised diagonal Gaussian as shown in equation (3.4),

$$\begin{aligned} q(Z) &= \prod_{u=1}^{m} q(z_u) & q(z_u|x_u) &= N(\mu_u, \mathrm{diag}(\sigma_u^2)) \\ p(z_u) &= N(0, I) & \forall u &\in 1, ..., m \end{aligned} \tag{3.4}$$

where I is the identity matrix. However, by assigning a different variational distribution for each user, the total number of parameters grows with the number of users. This significantly increases model complexity and could easily become the computational bottleneck in real-world applications. VAE resolves this issue by allowing all $q(z_u)$ to share the same parameterisation which is computed through a data-dependent function f_ϕ that takes the form of a neural network (often referred

as the inference network) with variational parameters ϕ.

$$q(z_u) = N\Big(\mu\Big(f_\phi(r_u)\Big), \text{diag}\Big\{\sigma^2\Big(f_\phi(r_u)\Big)\Big\}\Big) \quad \forall u \in 1, ..., m \tag{3.5}$$

It is worth noting that although the partially observed vector r_u is passed as the input into the inference network, the VAE is trained only on the observed entries x_u. Hence, only x_u appears in the objective function.

Unlike the variational distribution, the choice of $p_\theta(x_u|z_u)$ depends on the type of desired output distribution. Two slightly different versions of $p(x_u|z_u)$ are chosen to accommodate two types of prediction tasks. For feedback prediction, the model is evaluated for its ability to predict binary preferences. This is consistent with the assumption of a logistic log-likelihood, where each entry x_{ui} follows a Bernoulli distribution parameterised by $g(f_{ui})$. The logistic log-likelihood for user u is:

$$\log\, p_\theta(x_u|z_u) = \sum_i x_{ui}\, \log\, g(f_{ui}) + (1 - x_{ui})\, \log\,(1 - g(f_{ui})) \tag{3.6}$$

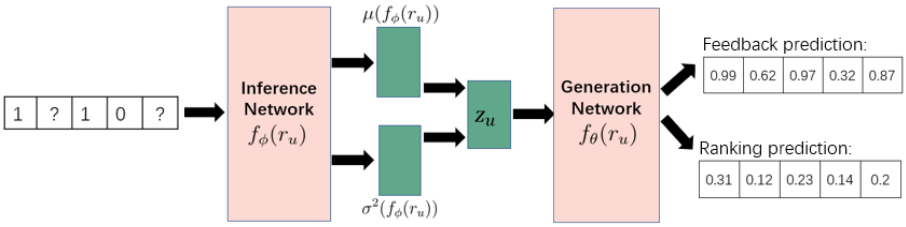
f_{ui} stands for $(f_\theta(z_u))_i$, where f_θ takes the forms of a neural network, parameterised by θ. The dependence on θ is made more explicit by subscripting $p(x_u|z_u)$ as $p_\theta(x_u|z_u)$. $g(\cdot)$ is chosen to be the sigmoid activation $(\frac{1}{1+\exp(-f_{ui})})$ that outputs the probability of item i being preferred by user u.

For ranking prediction, the model is tasked to generate a ranked list of items. This is consistent with the assumption of a multinomial log-likelihood, where the model is rewarded if it places higher probability masses on items that are preferred by the user. The multinomial log-likelihood for user u is:

$$\log\, p_\theta(x_u|z_u) = \sum_i x_{ui}\, \log\, g(f_{ui}) \tag{3.7}$$

To model the relationships among items, $g(\cdot)$ is chosen to be the softmax activation $(\frac{\exp(f_{ui})}{\sum_{i=1}^{n} \exp(f_{ui})})$ that outputs a probability distribution over items. It is worth noting that, since the current dataset solely contains binarised values, equation (3.6) and (3.7) only differ slightly, in that the former penalises the model if it places high probability masses on items with negative feedback, whereas the latter does not. Having this additional penalisation for negative feedback does not seem to be

Figure 3.1: VAE Model



contradictory to the objective of ranking prediction. Therefore, same as [4], the current study adopts the logistic likelihood in both types of prediction task for the sake of simplicity, although multinomial distribution should still be used if one was to generate a new rating matrix through ancestral sampling [2][1] The two variants of VAE are respectively named as VAE-sigmoid and VAE-softmax.

Having chosen the form of $p_\theta(x_u|z_u)$, we now need to compute its expectation $E_{q(z_u)}[\log p_\theta(x_u|z_u)]$. Following the common practice in the literature, the expected value is approximated with its unbiased estimate as follow:

$$E_{q(z_u)}[\log p_\theta(x_u|z_u)] \approx \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(x_u^l|z_u^l) \tag{3.8}$$

where z_u is sampled from $q(z_u)$. L is chosen to be one by convention.

Putting together what has been discussed so far, Fig. 3.1 illustrates the main working mechanism of a VAE model. The partially observed vector r_u is first passed through the inference network, which outputs the mean ($\mu(f_\phi(r_u))$) and variance ($\sigma^2(f_\phi(r_u))$) of the variational distribution. Then z_u is sampled from the resulting variational distribution, and fed as the input to the generation network. Sigmoid or softmax activation is then applied to the second last layer of the generation network to produce the probability of an item being preferred or a probability distribution over all items.

[1] Ancestral sampling refers to the process of generating new data points by first sampling variables with no parents from its prior distributions, then sampling another set of variables conditioned the sampled variables. The process continues until there is no variable left to be conditioned upon.

3.2.2 Reparameterisation Trick

Although the process outlined above constitutes the main bulk of the VAE model, a few additional modifications need to be made for it to be effective trained with back-propagation, which requires gradient computations with respect to two sets of parameters: ϕ and θ. Since KL divergence between two Gaussian is known in its closed form, the gradient of the KL term can be computed trivially. This is however, not true for the remaining term $-E_{q(z_u)}[log\, p_\theta(x_{ui}|z_u)]$. For θ, the gradient sign could be pushed into the expectation, which could then be approximated with its unbiased estimate as shown in equation (3.9). The same, however, could not be applied when taking the gradient with resepect to ϕ, as the expectation directly depends on it. The reparameterisation trick [9] offers a workaround for this issue through applying the change of variable as follows.

$$\begin{aligned} &\epsilon_u \sim N(0,1) \\ &z_u \sim N(\mu(f_\phi(x_u), \sigma^2(f_\phi(x_u))) \\ &z_u = \epsilon_u \odot \sigma(f_\phi(x_u)) + \mu(f_\phi(x_u)) \end{aligned} \tag{3.9}$$

$$\nabla E_{q(z_u)}[log\, p_\theta(x_u|z_u)] = E_{p(\epsilon)}[\nabla log\, p_\theta(x_u|\epsilon_u \odot \sigma(f_\phi(x_u))) + \mu(f_\phi(x_u)))] \tag{3.10}$$

Now the expectation is taken with respect to the new variable ϵ, the gradient sign could be again pushed inside the expectation, which leads to the final objective function:

$$\begin{aligned} \mathcal{L}^{VAE} &= \frac{1}{m}\sum_{u=1}^{m}\Big(- E_{p(\epsilon)}[\log\, p_\theta(x_u|z_u)] + KL[q(z_u)||p(z_u)]\Big) \\ &= \frac{1}{m}\sum_{u=1}^{m}\Big(- \sum_i \Big(x_{ui}\, \log\, g(f_{ui}) + (1-x_{ui})\, \log\,(1-g(f_{ui}))\Big) \\ &\quad + \sum_{k=1}^{K}\Big(\frac{1}{2}(2\, \log(\sigma_{uk}^2) - \mu_{uk}^2 - \sigma_{uk}^2 + 1)\Big)\Big) \end{aligned} \tag{3.11}$$

where $g(f_{ui}) = (f_\theta(z_u))_i$, $z_u = \epsilon_u \odot \sigma(f_\phi(x_u))) + \mu(f_\phi(x_u)$, $\epsilon_u \sim N(0,1)$.

3.2.3 Interpretation of $\mathcal{L}^{VAE}$ and Parameter β

A closer look at the objective function offers some important insights into the internal mechanism of the VAE model. The negative log-likelihood term prefers densities that place greater probability masses on the latent representations that captures the essence of the observed data, while the KL term encourages densities that closely resemble the prior $p(z)$, acting as a regularisation term. The $\mathcal{L}^{VAE}$ is often argued to be over-regularised for recommendation tasks, and introducing β that controls the trade off between reconstruction accuracy and prior constraint has been shown to lead to a superior prediction performance [13]. Therefore, following this convention, the $\mathcal{L}^{VAE}$ is modified as follows.

$$\begin{aligned}\mathcal{L}^{VAE} = \frac{1}{m}\sum_{u=1}^{m}\Bigg(&-\sum_{i}\Big(x_{ui}\,\log\, g(f_{ui}) + (1-x_{ui})\,\log\,(1-g(f_{ui}))\Big) \\ &+\sum_{k=1}^{K}\Big(\frac{\beta}{2}(2\,\log(\sigma_{uk}^2) - \mu_{uk}^2 - \sigma_{uk}^2 + 1)\Big)\Bigg)\end{aligned} \tag{3.12}$$

3.2.4 Incorporating Side Information

The current thesis adopts the same method as the one used in [1], where both ratings and auxiliary information are passed through the same VAE and encoded in the same latent variable. This could be achieved by transposing the $n \times d$ side information matrix Y to ensure the column dimension of Y and R is matched. Such a method is argued to alleviate the rating sparsity problem and has been shown to improve prediction performance when incorporated in the VAE model. Instead of using the bag-of-words representations of the textual review data like the authors did in [13], the current thesis uses one-hot encoded item category information.

Chapter 4

Experiment

4.1 Data Description and Prepossessing

The performance of the aforementioned models is evaluated using the 2019 Yelp Dataset Challenge [23]. Considering the limited computational resource, the current project limits its scope to restaurant recommendations in Quebec, Canada's largest state by area. Users with less than five ratings are removed from the dataset. Explicit ratings are binarised by setting the entries that are above 4 to be 1 and 0 otherwise. Since it is impossible to accurately assess the model performance on entries that are unknown, the models are only evaluated on the known entries by zero-masking a portion of which to be the test or validation set.

Two types of data-splitting are applied to accommodate prediction tasks at various difficulty levels. For the easy generalisation task, training, testing and validation sets are based on the same set of users and only differ in the positions of the zero-masked entries. The zero-masked entries for each user are selected randomly, with 10% of which being allocated to the validation set and another 10% to the test set.

For the harder generalisation task, the training, testing and validation sets are based on a completely different set of users. The ratings in the training set are kept intact, whereas, for the testing and validation set, 80% of the ratings are randomly allocated in the fold-in set with the remaining 20% being zero-masked

for model evaluation. The descriptive statistics for the datasets used in easy and hard generalisation task are shown in Table 4.1 and 4.2 respectively. Please note that the positive class ratio is only calculated for the observed entries.

Table 4.1: Descriptive Statistics for Easy Generalisation Task

	Train	Validation	Test
Number of Users	5536	5536	5536
Number of Items	5524	5524	5524
Sparsity	0.9980	0.9997	0.9997
Positive Class Ratio	0.6806	0.6783	0.6845

Table 4.2: Descriptive Statistics for Hard Generalisation Task

	Train	Validation		Test	
		Fold-in	Report	Fold-in	Report
Number of Users	4268	533	533	535	535
Numbr of Items	5524	5524	5524	5524	5524
Sparsity	0.9974	0.9987	0.9995	0.9986	0.9997
Positive Class Ratio	0.6810	0.6846	0.6773	0.6667	0.6680

4.1.1 Sampling Unreported Entries

There has been some longstanding challenges associated with dealing with implicit data, as it is often argued that while the positive feedback is observed with confidence, the nature of the missing entries is ambiguous, as it could be either negative or unobserved [6]. However, this might not be universally applicable to all types of implicit data. Although it is often true that a user's click records only contain information about the positive instances, the same does not apply to the binarised explicit ratings, where negative instances are also observed (the ones that are below 4). Hence, the conventional treatments of implicit data such as sampling a subset of zero entries as negative feedback or assigning lower confidence for negative feedback [16], are neither necessary nor applicable for binarised explicit ratings. Nevertheless, this does not mean the unobserved entries in the explicit setting are free of ambiguities. Consider any missing entry in a rating matrix, such an entry could represent one of the two scenarios: the user consumed the item but

did not provide any rating for it, or the user did not consume the item in the first place. Notice that only the latter scenario is truly non-informative, hence, the naive assumption that treats all missing entries as non-informative might not be applicable. However, this will cease to be an issue if the probability of the former scenario is negligible, which is likely to be the case for highly active users, i.e. users with strong rating habits. This is because, users of this type are more likely to express their preferences for an item after consumption, hence, decreasing the probability of a missing entry being consumed but not rated. However, this is unlikely to be true for users who rate occasionally, perhaps, only when the experience is surprisingly good or extremely bad. That is to say, the unobserved ratings could well be an unreported positive or a negative feedback, and the probability of this scenario is likely to increase for users with lower levels of activity. Hence, for the relatively inactive users, we tend to be less confident that the missing entries are truly uninformative. To account for such a possibility, for a given user u, c_u number of unobserved ratings are sampled as unreported ratings:

$$c_u = \begin{cases} \mu_\kappa - \kappa_u & \text{if } \kappa_u < \mu_\kappa \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

where κ_u is the rating count for user u, μ_κ is the mean rating count averaged over all users. The value of a sampled unreported rating r^*_{ui} is computed as the weighted average between item average rating (r^μ_i) and user average rating (r^μ_u):

$$\begin{aligned} r^*_{ui} &= \eta\, r^\mu_u + (1-\eta) r^\mu_i \\ \eta &= \begin{cases} 0.8 & \text{if } c_u < 4 \\ 0.2 & \text{otherwise} \end{cases} \end{aligned} \tag{4.2}$$

η is chosen such that a higher weight is assigned to the item mean if the user is considered relatively inactive. It is important to note that the average rating given as part of the Yelp dataset is computed based on the entire rating history, hence, should not be used for r^μ_u and r^μ_i to prevent data leakage.

4.2 Evaluation Scheme

Model performance of MF and VAE-sigmoid is compared using the easy generalisation task. The harder generalisation task is only employed for the VAE models as MF inevitably requires a certain form of re-training to learn the latent representations for a new set of users, which leads to an unfair comparison with its non-linear counterparts.

4.2.1 Evaluation Metrics

Since the explicit ratings are binarised into zero and ones, predicting whether a customer will be interested in certain items or not is essentially translated into a classification problem. Two popular classification metrics are shown as follow.

$$\text{recall} = \frac{\#\text{True Positives}}{\#\text{ True Positives} + \#\text{False Negatives}} \tag{4.3}$$

$$\text{specificity} = \frac{\#\text{True Negatives}}{\#\text{ True Negatives} + \#\text{False Positives}} \tag{4.4}$$

Although the two above metrics could accurately measure how well the model is at identifying relevant and irrelevant items, it could be argued that, in practice, the goal of a RS is not to exhaustively identify all the items a user might or might not like. Instead, recommendations are often generated by selecting a few items from a pool of items that are predicted to be relevant. In this sense, a RS system is considered effective if it is able to output a reasonably large pool of recommendation candidates, such that an item randomly selected from this pool will be liked by the users with a relatively high probability. Performance of this type is assessed by another metrics: precision, which is computed as follow:

$$\text{precision} = \frac{\#\text{True Positive}}{\#\text{True Positive} + \#\text{False Positive}} \tag{4.5}$$

For ranking prediction tasks, the ranking measure recall@$\mathcal{N}$ is used to evaluate the model's ability to identify the positively rated items in a ranked list of length $\mathcal{N}$ ($\mathcal{N} << n$). This is formally defined as follow:

$$recall@N = \sum_{u=1}^{m} \frac{\sum_{r=1}^{N} I[\omega(r) \in B_u]}{min(\mathcal{N}, |B_u|)} \tag{4.6}$$

where $I[\,]$ is the indicator function that returns one if the statement inside $[\,]$ is true and zero otherwise, $\omega(r)$ is the rth item in the sorted recommendation list and B_u is the set of positively rated items for user u. To put it more simply, the numerator of equation (4.6) is computed by selecting the top $\mathcal{N}$ items from a given ranked list and then counting how many of the $\mathcal{N}$ items are actually rated as positive by user u. Consistent with the common practice in the literature, $\mathcal{N}$ is set to be 20, 50, and 100, although it should be noted that a smaller $\mathcal{N}$ is generally a better indication of the model performance, as in practice it is rarely the case that users will be presented with a list of 100 recommended items all at once.

It is important to note that recall@$\mathcal{N}$ requires the predicted items to be ranked according to a certain criteria, which could be achieved by sorting the output of the VAE-softmax (i.e. a probability distribution over items) and selecting the top $\mathcal{N}$ items to be the recommended restaurants. However, this is not possible for MF and VAE-sigmoid, as the predicted entries are binarised into zeros and ones, providing no information about the relationships among items. The following table summarises the type of evaluation metrics and prediction task used for each model.

	Easy Generalisation Task	Hard Generalisation Task
Recall,Specificity,Precision	VAE-sigmoid, MF	VAE-sigmoid
Recall@$\mathcal{N}$	N/A	VAE-softmax

4.3 Implementation Details

4.3.1 Matrix Factorisation via Alternating Minimisation

Recall that the objective function for the MF model is given as follow:

$$\mathcal{L}^{MF}(P,Q) = -\sum_{i=1}^{r}\Bigg(P_{\Omega}\Big(R \odot \log \varsigma(PQ) + (J-R)\odot \\ log\left(J - \varsigma(PQ)\right)\Big)\Bigg)_i + \frac{\lambda}{2}(\|P\|^2 + \|Q\|^2) \tag{4.7}$$

The Alternating Minimisation algorithm solves the original non-convex optimisation problem by alternating between minimising $\mathcal{L}^{MF}(P,Q)$ with respect to P and Q, which are learned through gradient descent. The gradient updates for P and Q computed as follow.

$$\begin{aligned} &\nabla L_P(Q^k) = P^T P_{\Omega}^T[P_{\Omega}(\sigma(P^k Q^k)) - P_{\Omega}(R)] + \lambda Q^k \\ &Q^{k+1} = Q^k - \tau \nabla L_P(Q^k) \end{aligned} \tag{4.8}$$

$$\begin{aligned} &\nabla L_Q(P^k) = P_{\Omega}^T[P_{\Omega}(\sigma(P^k Q^{k+1})) - P_{\Omega}(R)](Q^{k+1})^T + \lambda P \\ &P^{k+1} = P^k - \tau \nabla L_Q(P^k) \end{aligned} \tag{4.9}$$

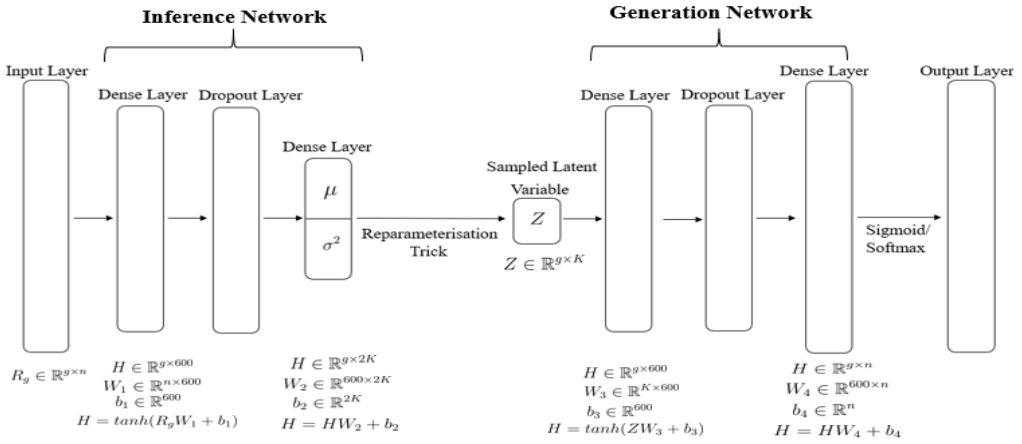
P and Q are initialised with standard Gaussian distribution ($N(0, I)$), where I is the identity matrix. Training is terminated when the validation precision stops increasing in three consecutive iterations. After training is completed, the reconstructed R^* is binarised by setting the values above 0.5 to be 1 and 0 otherwise.

4.3.2 VAE

The overall architecture of the VAE model is shown in Fig. 4.1. Input vector is passed through a 600-unit hidden layer with Tanh activation. No activation function is applied to the last inference layer, as the output of inference network is directly treated as the mean and variance of a Gaussian random variable. The dummy variable ϵ is then sampled from its prior to compute the latent variable

via the reparameterisation trick. The sampled latent variable is then fed as the input into the generation network which has an architecture that is symmetrical with the inference network. The sigmoid or softmax transformation is applied to the second last layer in the generation network depending on the variants of VAE implemented. Predictions are generated by first computing a forward pass through the inference network. The resulting mean of the variational distribution is then directly used as the input to the generation network. For VAE-sigmoid, a predicted entry is set to be one if its value is above 0.5 and zero otherwise. For VAE-softmax, the output is sorted in a descending order to produce a ranked item list. To incorporate side information, the $d \times n$ one-hot encoded item category matrix Y is first passed as input at the pre-training stage. Then VAE is initialised with the pre-trained weights to learn the latent features of R.

Figure 4.1: Network Architecture



Stopping Criteria and the reweighing scheme

To decide on which measure should the stopping criteria be based upon, the current study has first conducted a few experiments where training is terminated at the peak of the validation recall. Results of such experiments suggest that this could lead to predictions that are strongly biased towards the positive class, with a below initialisation specificity. An alternative approach is to stop at the peak of the precision. This could potentially overcome the issue as the precision not only

encourages true positives but also penalise predictions that contain a high number of false positives. Although the positive prediction bias is only observed for the VAE model, the same stopping criteria is also adopted for the MF model to ensure a fair model comparison.

It should be note that stopping at the peak of the precision only offers a workaround for the biased prediction problem, it does not however, directly eliminate the problem. A reweighing scheme is therefore, proposed to address this issue by reweighing the positive and negative term in $log\, p_\theta(x_u|z_u)$ by coefficient α:

$$\begin{gathered} \alpha : (1-\alpha) = t_p : t_n \\ E_{p(\epsilon)}[\log\, p_\theta(x_{ui}|z_u)] = \sum_i (1-\alpha) x_{ui}\, \log\, g(f_{ui}) + \alpha\, (1-x_{ui})\, \log\, (1-g(f_{ui})) \end{gathered} \tag{4.10}$$

t_n and t_p respectively represents the number of negative and positive entries in the training set. Such a reweighing scheme is proposed based on the speculation that the class imbalance in the training set (as shown in table 4.1 and 4.2) might have contributed to the observed prediction bias. The effectiveness of the reweighing scheme primarily lies in its ability to indirectly counter such a class imbalance by reweighing the log-likelihood such that the positive to negative ratio restores to 1:1.

4.3.3 Hyperparameter Tuning

The Tree-structured Parzen Estimator from the python hyperopt package is used for hyper-parameter tuning. The hyperparameters and their corresponding sample space for each model are listed in the table 4.3. A total of 100 evaluation trials is conducted for each model. The hyper-parameters for MF and VAE are tuned on the easy generalisation task. No sampling scheme nor side information is incorporated for any model at this stage. Ideally, the hyper-parameter search should be conducted for every set of experimental conditions considered. However, this could not be achieved in reasonable time given the limited GPU resource.

Table 4.3: Hyperparameters for VAE and MF model

VAE	MF
$\tau \sim U(0.0001, 0.001)$	$\tau \sim U(0.001, 0.01)$
$\beta \sim U(0.6, 0.8)$	$\lambda \sim U(0.1, 0.7)$
$K \sim U(5, 50)$	$K \sim U(20, 50)$
$\rho \sim U(0.1, 0.3)$	

4.3.4 Experiment Outlines

Since the current study contains quite a few sets of experiments pertinent to our analysis, it is useful to clarify how experimental comparisons are carried out. Recall that the four main research goals of the current study are as follows:

1. Examine the effect of reweighing scheme on VAE-sigmoid for rating prediction task
2. Explore the effect of sampling unreported entries on rating and ranking prediction task
3. Compare the performance of VAE and MF on rating prediction task
4. Examine the effect of incorporating side information on rating prediction task

Table 4.2 illustrates the experiments conducted for each research goal and the order in which they are carried out.

Although a higher metric score is often interpreted as an indicator for superior model performance, there is a possibility that the observed improvements occurred to due to chance, which means the same patterns might not be replicated if a different datset was to be used. To ensure the reliability of our conclusions, per user significance testing is performed on the results obtained from all experimental comparisons. Computation of the user-level metrics score is likely to result in some cases where the denominator of the evaluation metrics is zero. These entries will result in NaN values and are therefore, dropped from the evaluation. Since the denominators of specificity, recall and recall@$\mathcal{N}$ are computed based on the ground truth, the set of non-NaN users is same for all models considered. Therefore, paired sample t-test is used to examine if the observed differences in mean user-level

metrics scores are statistically significant. However, for precision, the denominator depends on the number of predicted positives which is likely to differ from model to model. Therefore, independent sample t-test is used instead, to account for the different set of non-NaN users.

Figure 4.2: Experiment Outline

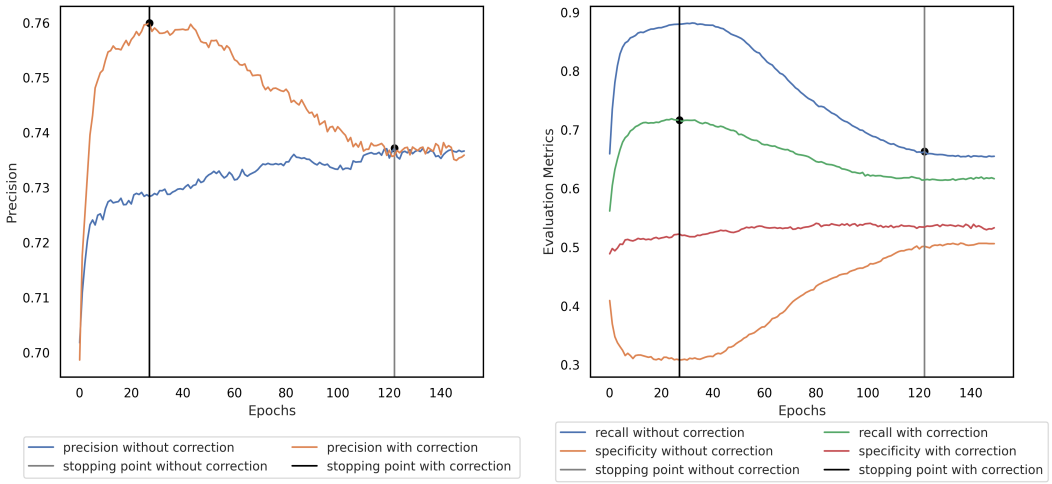
Research Goal 1	Easy Geneneralisation	VAE-sigmoid with Reweighing	VS.	VAE-sigmoid without Reweighing
Research Goal 2	Easy Geneneralisation	VAE-sigmoid without Reweighing	VS.	MF
		VAE-sigmoid with Reweighing	VS.	MF
Research Goal 3	Easy Geneneralisation	VAE-sigmoid without Sampling	VS.	VAE-sigmoid with Sampling
	Hard Geneneralisation	VAE-sigmoid without Sampling	VS.	VAE-sigmoid with Sampling
		VAE-softmax without Sampling	VS.	VAE-softmax without Sampling
Research Goal 4	Easy Geneneralisation	VAE-sigmoid with Side Information	VS.	VAE-sigmoid without Side Information
	Hard Geneneralisation	VAE-softmax with Side Information	VS.	VAE-softmax without Side Information

4.4 Experiment Result

4.4.1 Research Goal 1: Effect of the Reweighing Scheme

Figure 4.3 illustrates the training trajectories of the VAE-sigmoid model with and without reweighing. All statistics are reported on the validation set. As mentioned previously, when no reweighing is applied, the recall peaks at an extremely high value, with specificity simultaneously dropped to its minimum. Recommendations generated at this point are largely uninformative, as the system is predicting the majority of the restaurants to be positively rated, which is contrary to the goal of an information filtering tool that the system is designed for. Although stopping at the peak of the precision has led to a much lower recall and a much higher specificity, the problem of prediction bias still remains. This has, however, been

Figure 4.3: Effect of the Reweighing Scheme on Validation Set

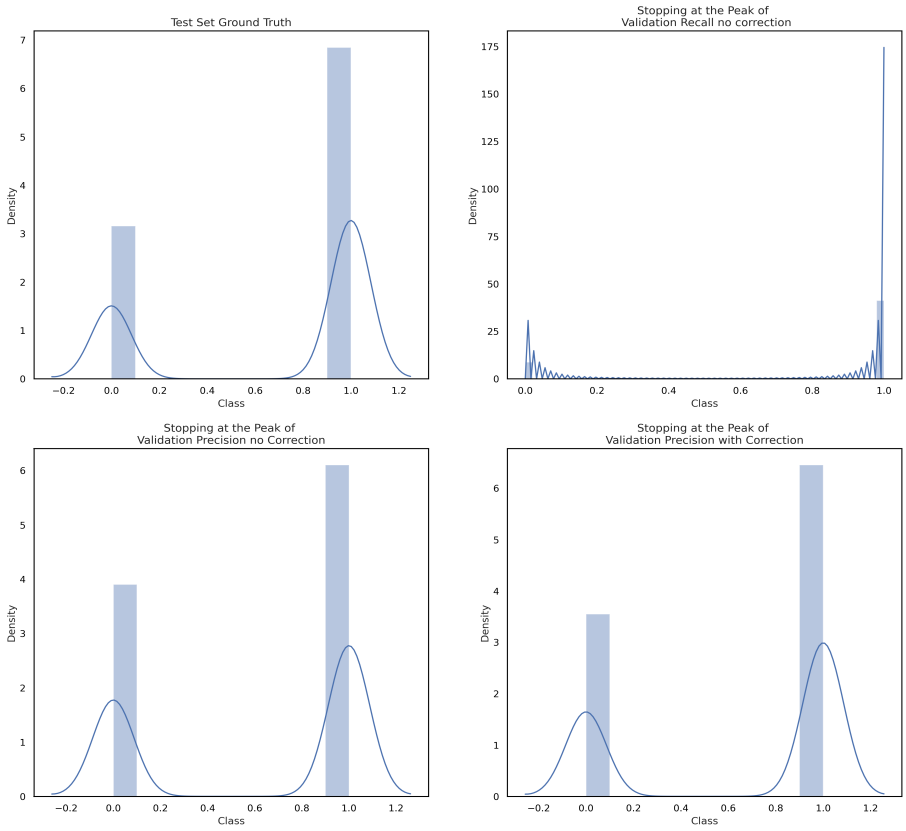


shown to be effectively dealt with by the proposed reweighing scheme. The initial drop in specificity is no longer observed, allowing the peaks of the precision and recall to be more closely aligned.

The bias-correcting effect of the stopping at the peak of validation precision and reweighing scheme is also evident in the figure 4.4. Although the predictions no longer concentrate at the positive class for both cases, the class distribution more closely resembles the ground truth distribution when reweighing is applied.

Although the proposed reweighing scheme seems to be working as expected, it does not offer a definite explanation for the reason underlying the positive class bias, as the unweighted cross-entropy loss should have penalised the reconstruction errors for the negative feedback in the first place. Therefore, despite its more optimal performance, it is not clear as to whether the result the reweighing scheme generates represents the 'true' converged result, especially given its corresponding training reconstruction accuracy is much lower (77% compared to 99% when no reweighing is applied). Therefore, results from both stopping points are used when comparing to the MF model to ensure a fair comparison. Whereas, for experiments that are solely based on VAE-sigmoid models, only the one with reweighing is applied. There is however, one exception. Since models that are pretrained with

Figure 4.4: Class Distribution for VAE-sigmoid with Various Stopping Points



side information do not appear to have suffered from the same positive prediction bias, training is terminated at the peak of recall with no reweighing applied. A more detailed discussion of such a phenomenon could be found in subsection 4.4.4 .

The above plots are produced using the easy generalisation task. The same pattern is also found for the hard generalisation task which is presented in figure A.1 in the appendix.

4.4.2 Research Goal 2: VAE-sigmoid VS. MF

The current section discusses the results for MF and VAE-sigmoid models on easy generalisation task without sampling. For reasons mentioned in the previous

section, two sets of results are presented for the VAE-sigmoid models. High-level model comparisons are first carried out based on three test set evaluation metrics. Then, a more fine grained analysis of the model performance is presented by examining the training trajectory of the four additional metrics: True Positives (TP), True Negatives (TN), False Positive (FP) and False Negative (FN).

Table 4.4: Model Comparison

	MF	Unweighted VAE-sigmoid	Weighted VAE-sigmoid
recall	0.5375	0.6387***	0.7139***
specificity	0.4723	0.5016	0.5107*
precision	0.6970	0.7465***	0.7689***

*** and *** respectively indicates statistical significance at $\alpha = 0.05$ and $\alpha = 0.001$ when compared to the linear baseline using paired sample t-test**

As shown in table 4.4, both versions of the VAE-sigmoid models have outperformed the MF model for all metrics measured by a statistically significant margin, except for the specificity generated by the unweighted VAE-sigmoid model. This implies that the VAE-sigmoid model is generally better at identifying the relevant items and avoiding recommending items that are considered irrelevant. The magnitude of the performance gap is shown to be more prominent for the weighted VAE than its unweighted version.

Figure 4.5: Validation Metrics during Training (VAE VS. MF)

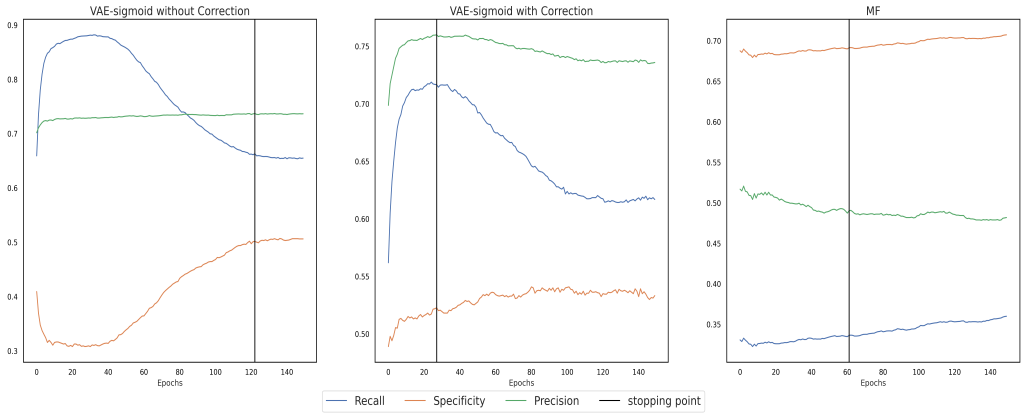
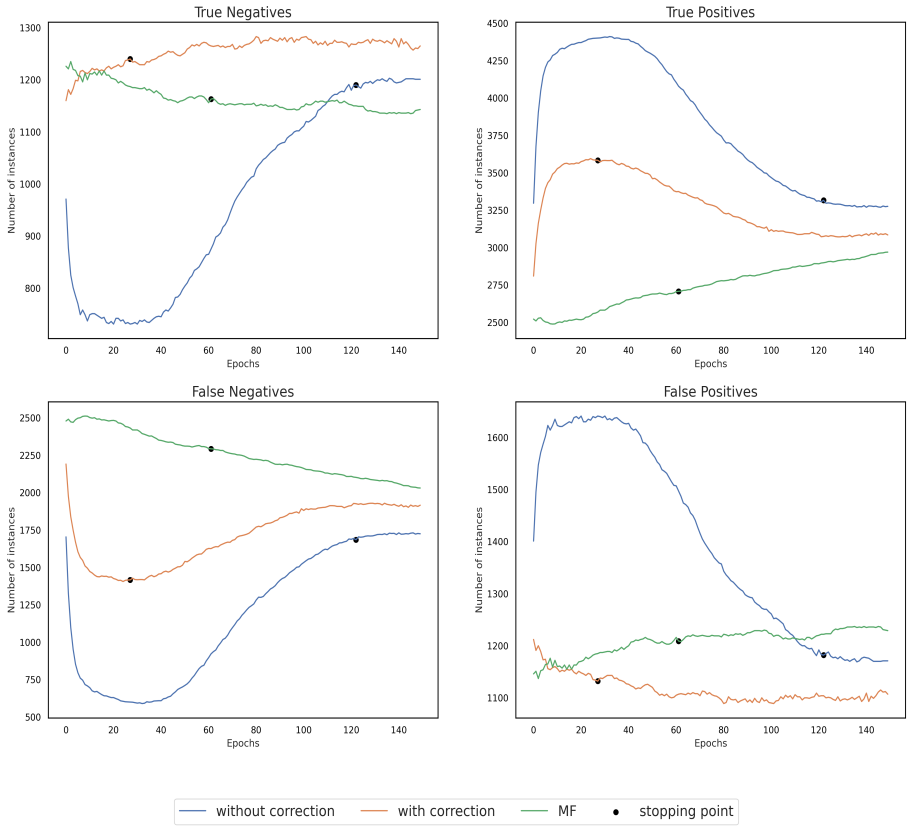


Figure 4.6: Trajectories of TN,TP, FN, FP during Training (VAE VS. MF)



A closer examination of the training trajectories of each implemented model has revealed some further insights. As shown in figure 4.5, the recall for the VAE-sigmoid model is initialised at a much higher value than that of the MF model. Although such a difference could have potentially contributed to the superior performance of the VAE model, further experiments have shown that this does not seem to be the case, as its validation recall has plateaued at approximately the same level even when the value at the initialisation is approximately the same as the one for MF. Visualisations of the additional experiment could be found in figure A.2 of the appendix.

Although the recall for the MF model has increased as the training proceeds, the specificity has experienced a noticeable drop and ended up being much lower

than the value at initialisation. Such an issue is more evidently shown in figure 4.6, where the number of FP has almost been continually increasing while the number of TN has dropped below initialisation. This implies that an increasingly large amount of errors has been made among items that are predicted as irrelevant, while at the same time, the truly irrelevant items are less likely to be identified. The unweighted VAE-sigmoid model also suffers from a similar issue but to a lesser extent, as the specificity did eventually increase back to the initialisation level. With reweighing, the evaluation metrics follows a completely different training trajectory, where the specificity has increased almost continuously throughout training. Although such an improvement has illustrated the effectiveness of the proposed reweighing scheme, the scheme is, however, not without limitations. Despite the fact that specificity has now followed a much more acceptable trajectory, the improvement from its baseline is still quite negligible, which could be attributed to a relatively slow increase in the number of TN and slow decrease in the number of FP. However, this might not be as problematic as it seems, since TN is often not considered to be as an important target in the context of a recommendation task, where recommended items are generated based on a pool of predicted positives not otherwise. Therefore, it is reasonable to conclude that overall, the weighted VAE-sigmoid has yielded satisfactory performance, and when compared against with the MF model, the results provide some direct empirical evidence for the frequent claim regarding the superior performance of AE-based model over its linear counterpart.

4.4.3 Research Goal 3: Effect of Sampling

The current section summarises the effect of sampling for VAE-sigmoid and VAE-softmax. Results of the two model variants are presented separately as follow.

VAE-sigmoid

As shown in table 4.6, although sampling unreported entries has improved the precision and specificity on both types of datasets, only the improvement in specificity has reached statistical significance. The recall follows a slightly different pattern,

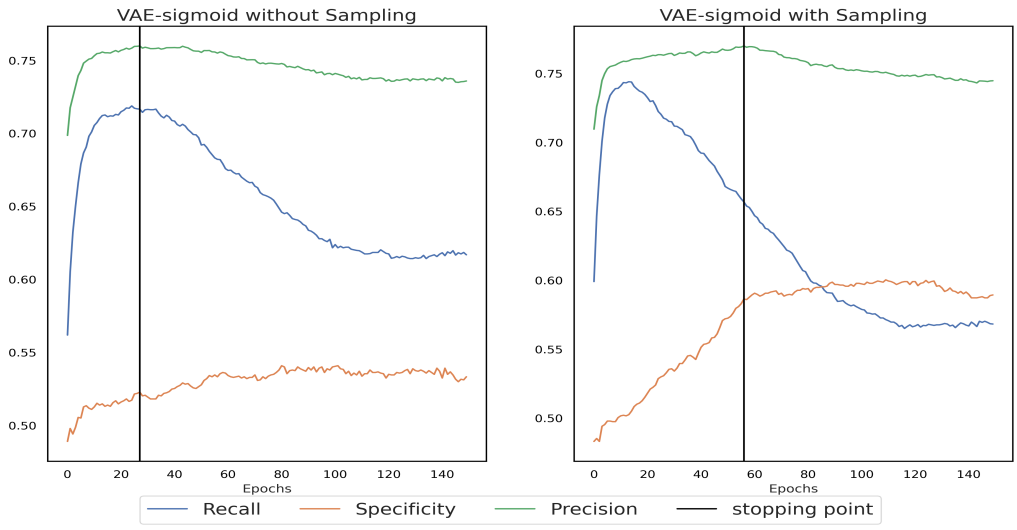
Table 4.6: Effect of Sampling for VAE-sigmoid

	Easy Generalisation Task		Hard Generalisation Task	
	No Sampling	Sampled	No Sampling	Sampled
recall	0.7140	0.6641***	0.8149	0.7516***
specificity	0.5107	0.5713***	0.5763	0.6457*
precision	0.7689	0.7797	0.7898	0.8202

***, *** respectively indicates statistical significance at $\alpha = 0.05$ and $\alpha = 0.001$ when compared to the unsampled counterpart**

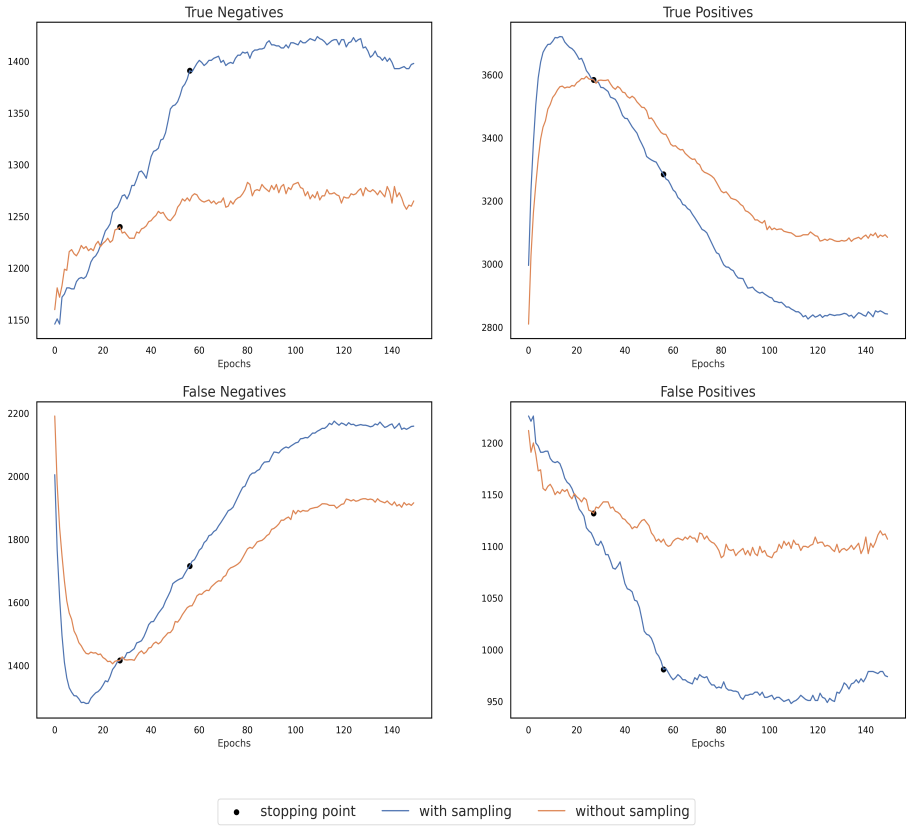
where the performance has dropped by a statistically significant amount for tasks at both difficulty levels after sampling is applied. This suggests that sampling unreported entries has led the model become better at filtering out irrelevant items at the expense of making more mistakes in identifying the relevant ones. When considering the trade off between the two, (i.e. the precision), the overall impact of sampling has been positive but at a negligible magnitude. In spite of this, the precision of the sampled VAE model is still considered rather promising, especially given that the Yelp Dataset has been shown to be harder than other commonly used datasets (e.g. Amazon) [5].

Figure 4.7: Validation Metrics during Training (Effect of Sampling)



To further understand the contributing factors of the observed changes in the

Figure 4.8: Trajectories of TN,TP, FN, FP during Training (Effect of Sampling)



three performance measures, we again examine their respective training trajectories along with four additional metrics: FN, FP, TP, TN. As shown in figure 4.8, contrary to the expectation, the peak of TP and the lowest point of FN are both at a better position for the sampled VAE-sigmoid, which means its performance in recall could have been more superior than the unsampled version if the training was to be terminated at the peak of the validation recall. However, since we are also considering the FP as an important metric, training is terminated at a much later point which led to a noticeable drop in recall.

Regarding the two contributing elements of specificity: TN and FP, the sampled version is at a worse position in the first few epochs but has quickly outperformed the unsampled version by a significant margin from 20 epochs onwards. The

greater reduction in the number of FP implies that the recommended restaurants are less likely to be irrelevant. The practical implications of such an improvement differ depending on the type of stakeholder considered. For consumers, this could be especially beneficial, as it directly reduces the amount of attentional resources wasted from browsing irrelevant information. Whereas for restaurant owners, unlike FN, the economic cost of FP is much lower, as being recommended to an incorrect audience is unlikely to incur any revenue loss.

Please note that the above plots are generated using easy generalisation task, the same claim also applies to the results produced using hard generalisation task, which are illustrated in figure A.3 and A.4 of the appendix.

VAE-softmax

As shown in table 4.8, although sampling unreported entries has led to a slight deprecation in performance for $\mathcal{N}$ at all length considered, only the one for $\mathcal{N} = 100$ has reached statistical significance. It is possible that the ranking prediction task is slightly less sensitive to the class distribution in the training set than the rating prediction task, as 10% decrease in the positive class ratio introduced by the sampling scheme has made a much less noticeable impact on the model performance. Overall, sampling unreported entries does not appear to be an effective strategy for improving ranking quality on the current dataset. When considering the VAE-softmax model alone, its performance is very close to the current state of the art (Multi-VAE), but it should be noted that a different dataset was used for the state of the art model [13].

Table 4.8: Effect of Sampling for VAE-softmax

	Without Sampling	With Sampling
Recall@20	0.3441	0.3169
Recall@50	0.4394	0.4164
Recall@100	0.5232	0.4847*

*** indicates statistical significance at $\alpha = 0.05$ when compared to the unsampled counterpart**

4.4.4 Research Goal 4: Effect of Side Information

The current section examines the effect of side information on VAE-sigmoid and VAE-softmax. Results of the two sets of experiments are respectively produced using easy generalisation task and hard generalisation task.

Table 4.10: Effect of Side Information for VAE-sigmoid on Easy Generalisation Task

	Without Side Information	With Side Information
Recall	0.7140	0.6511***
Specificity	0.5107	0.5190
Precision	0.7689	0.7606

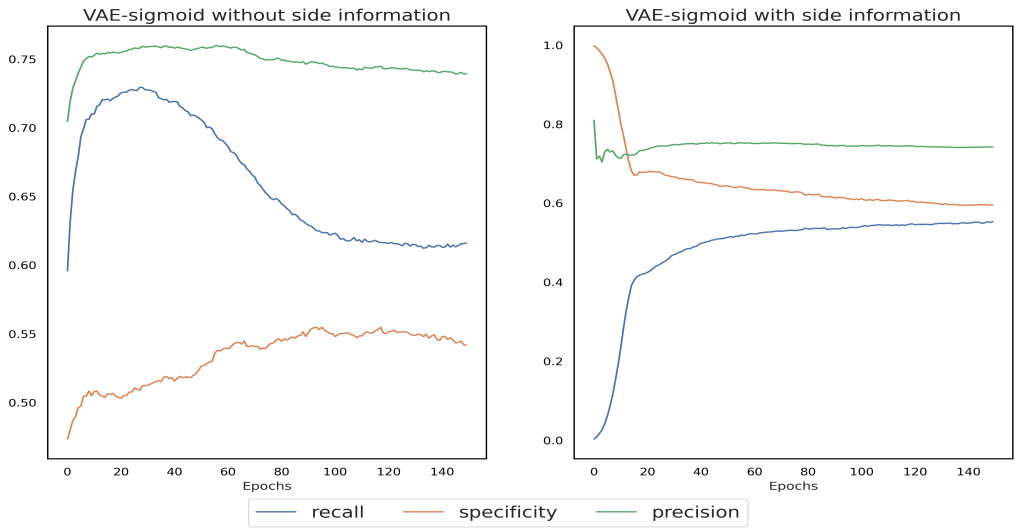
**** indicates statistical significance at $\alpha = 0.001$ when compared to the model without side information using paired sample t-test**

As shown in table 4.10, the addition of side information has led to a statistically drop in recall. Although precision and specificity has respectively shown a slight decrease and increase, the magnitude of such changes has not reached statistical significance. This implies that with pretraining, the model has become worse at identifying the relevant restaurants, while when taking into account to its ability to filter out irrelevant restaurants, its overall impact has been relatively neutral.

A closer inspection of the training trajectory of the three reported metrics has shown that with side information included, the specificity is initialised at a much higher level and has been continuously dropping throughout training. As shown in table 4.9 although the precision has eventually started increasing, its value has still remained below initialisation, which consequently prevents us from using it as the basis metrics for stopping criteria. It is also worth noticing that, the positive prediction bias has disappeared, with the recall peaking at a much lower value.

By further breaking down the three metrics into its respective constituents, figure 4.10 has shown that when side information is incorporated, the number of TP and FP is almost around zero at initialisation, while the number of TN and FN is much higher than its counterpart, suggesting that the model is now initialised at a point where the majority of the items are predicted to be irrelevant. Such results is indeed consistent with the data distribution in the side information matrix, where 99.26% of the entries are zeros. Initialisation does not appear to be

Figure 4.9: Validation Metrics during Training (Effect of Side Information)



the only problem arises from the addition of the side information, the number of FP has almost continuously increased throughout training while the number of TN has dropped significantly. This suggests that the model is becoming increasingly worse at identifying the irrelevant restaurants while making more mistakes in its recommendation lists.

A similar performance deprecation has been found for VAE-softmax when evaluated on the hard generalistion task. As shown in table 4.12, the recall evaluated at all $\mathcal{N}$ dropped by a statistically significant amount after side information is incorporated. This is also evident in figure 4.11, where although, the recall@20 of VAE-softmax^{side} [2] increases much faster in the beginning, it also starts dropping much earlier on during training, leading to a lower value at termination. Therefore, overall, results from the current study suggests that incorporating item category information through pretraining is not effective in improving recommendation quality.

[2]VAE-softmax with side information incorporated

Figure 4.10: Trajectories of TN,TP, FN, FP during Training (Effect of Side Information)

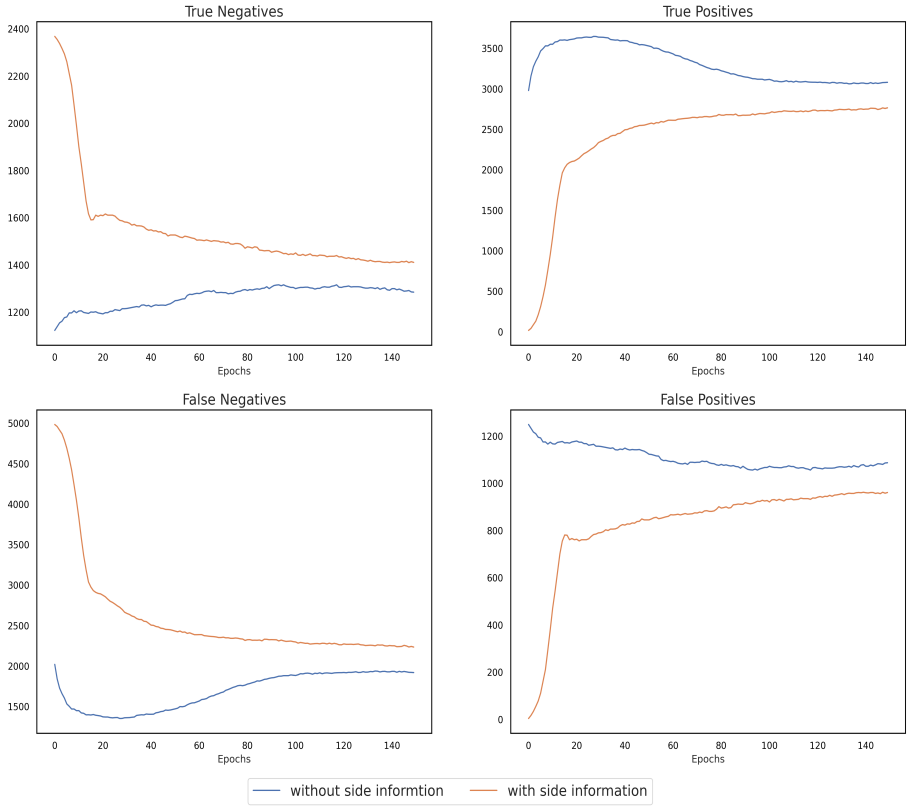
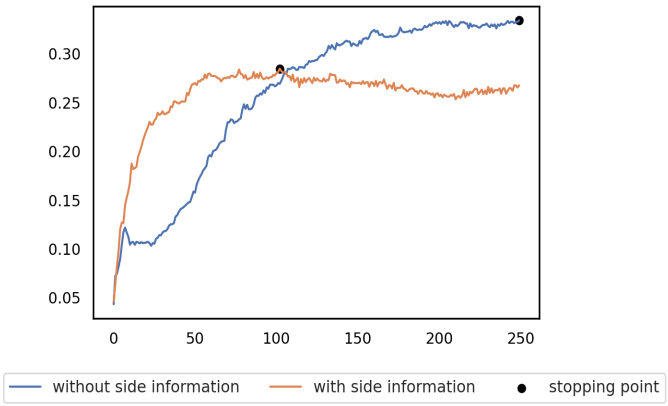


Table 4.12: Effect of Side Information for VAE-softmax on Hard Generalisation Task

	Without Side Information	With Side Information
Recall@20	0.3441	0.2981*
Recall@50	0.4394	0.3801**
Recall@100	0.5232	0.4512***

***, ** and *** respectively indicates statistical significance at $\alpha = 0.05$, $\alpha = 0.005$ and $\alpha = 0.001$ when compared to the model without side information using paired sample t-test**

Figure 4.11: Recall@20 during Training for VAE-softmax (Effect of Side Information)



Chapter 5

Conclusions

The current study has contributed to the literature by introducing a reweighing scheme that is shown to be effective in eliminating the positive class bias for VAE-sigmoid models when evaluated using generalisation task at various difficulty levels. The study has also provided a level of empirical support for the frequent claim regarding the superior performance of the AE-based models over its linear counterpart. In addition, the alternative sampling scheme proposed for binarised explicit data has also produced promising results in helping the model to become better at filtering out irrelevant items. The addition of side information, however, did not improve the model performance and has led to some noticeable performance deprecation in most metrics considered. This might due to the high sparsity of the item category matrix and the manner in which side information is integrated.

Despite the insights that the current study provides, the aforementioned results should be interpreted with caution, as the study also suffers from several limitations. The major one is the potential bias that is inherent in the data collection process. This is primarily due to the fact that results produced from the current experiments are only based on the observed entries which constitute less than one percent of the entire dataset. In practice, a RS is considered effective if the recommended items are considered as relevant by the users. Feedback of this type is not necessarily equivalent to the observed ratings, as it could well be the case that a user considers a recommended item as relevant and eventually visits

the restaurant, but decides not to report such an experience onto the system. This type of feedback theoretically still counts as evidence for the effectiveness of an RS but is failed to be captured by the current evaluation method. Although a sampling scheme has been proposed to account for the possibility of such a scenario, the scheme was only applied to the training set not to the validation and test set, which means our conclusions are still drawn from results that are solely based on the observed entries. This could be problematic given that the observed entries of the current dataset has a noticeable bias towards the positive feedback, which leaves the possibility that users are more likely to rate items that they had a positive experience with. Such an observation is consistent with the findings produced by Marlin et al.[14] who demonstrated that users' decisions on whether or not to rate an item is directly confounded by their preference for the item, which again highlights the distribution gap between observed and unobserved entries. Since the proposed sampling scheme could potentially close such a distribution gap by re-adjusting the positive to negative class ratio in the training set, it is possible that performance enhancing effect of the sampling scheme would be more prominent when evaluated at deployment time. Such observations also highlight the essential role that the online user experiments play in producing results that generalise beyond the immediate context of the experiments.

Another limitation that most studies in the recommendation literature, including ours, suffer is its generalisability to users with fewer than five ratings, since users of this type is typically dropped by convention. However, in practice, recommendations need to be generated for all users regardless of their activity levels. This might lead to a level of performance drop since inclusion of all users will inevitably increase the sparsity of the feedback matrix. The third limitation is related to the choice of evaluation metrics. Although a comprehensive range of metrics are used for the feedback prediction task, only one is adopted for the ranking prediction task. In future research, additional metrics such as Truncated Normalized Discounted Cumulative Gain could also be used to gain a deeper understanding of the quality of recommendation list by accounting for the relative positions of the ranked items. On top of this, further experiments that attempt to replicate such results on other datasets are also recommended for better generalisability.

Appendix A

Appendix

To reproduce the statistics presented in the result section, please download the trained weights through the link here. The function used to compute recall@$\mathcal{N}$ is in part borrowed from [12]. Please find below the additional plots that are referred in the result section.

Figure A.1: Evaluation metrics during Training for VAE-sigmoid (Effect of Reweighing Scheme)

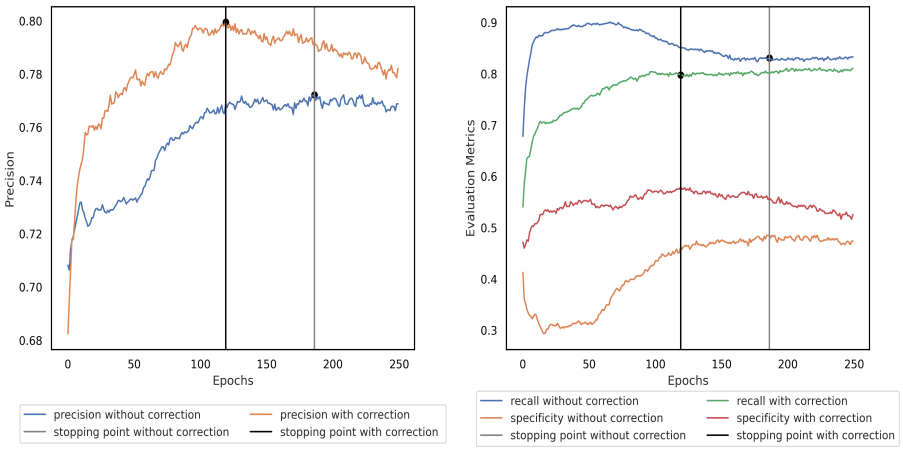


Figure A.2: Trajectories of Evaluation Metrics of VAE-sigmoid with Different Initialisations

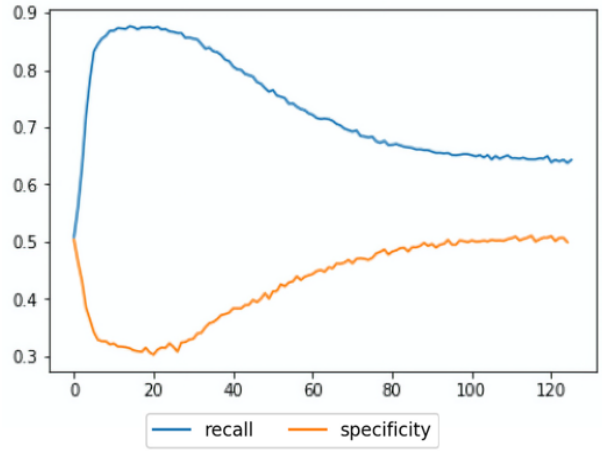


Figure A.3: Evaluation Metrics During Training for VAE-sigmoid (Effect of Sampling)

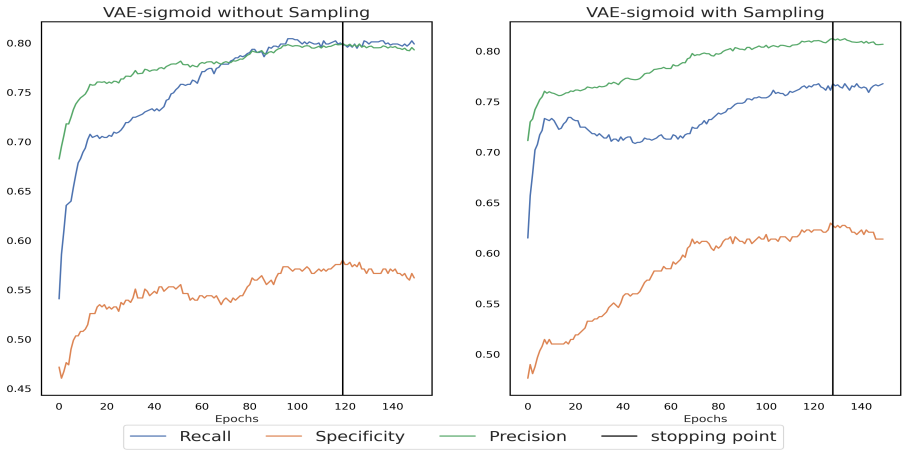
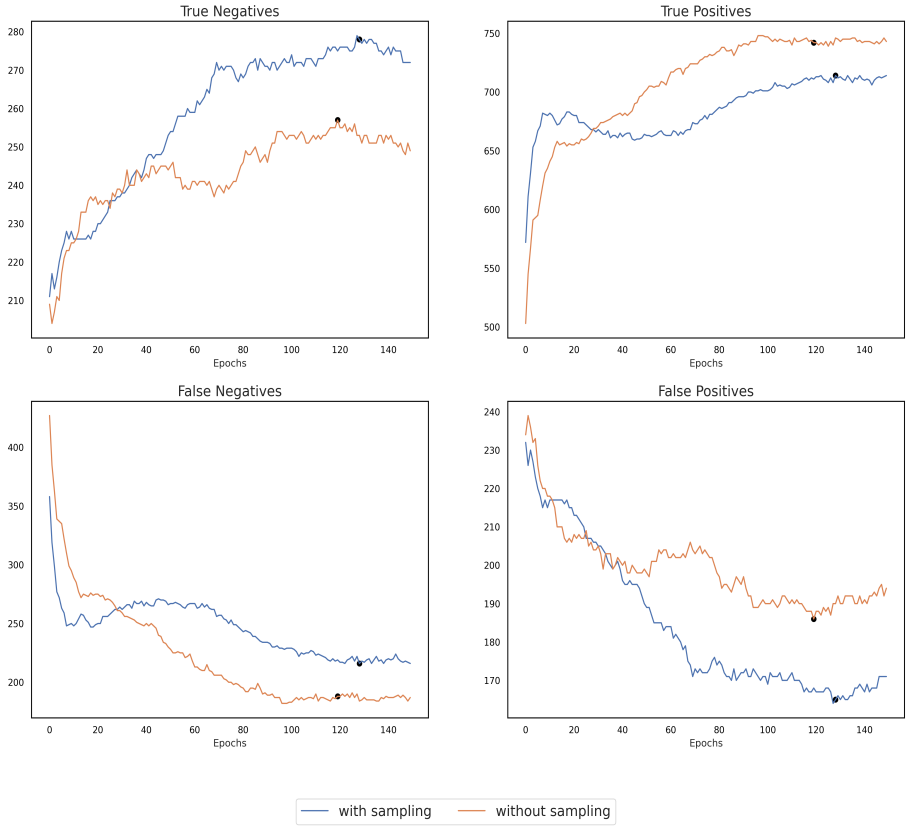


Figure A.4: Trajectories of TN, TP, FN, FP during Training (Effect of Sampling)



Bibliography

[1] Chen, Y., and de Rijke, M. A collective variational autoencoder for top-n recommendation with side information. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems* (2018), pp. 3–9.

[2] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine 35*, 1 (2018), 53–65.

[3] Doersch, C. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908* (2016).

[4] Gupta, K., Raghuprasad, M. Y., and Kumar, P. A hybrid variational autoencoder for collaborative filtering. *arXiv preprint arXiv:1808.01006* (2018).

[5] He, X., Zhang, H., Kan, M.-Y., and Chua, T.-S. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016), pp. 549–558.

[6] Hu, Y., Koren, Y., and Volinsky, C. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining* (2008), Ieee, pp. 263–272.

[7] Jain, S., Grover, A., Thakur, P. S., and Choudhary, S. K. Trends, problems and solutions of recommender system. In *International conference on computing, communication & automation* (2015), IEEE, pp. 955–958.

[8] Jawaheer, G., Szomszor, M., and Kostkova, P. Comparison of implicit and explicit feedback from an online music recommendation service. In *proceedings of the 1st international workshop on information heterogeneity and fusion in recommender systems* (2010), pp. 47–51.

[9] Kingma, D. P., and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[10] Lee, W., Song, K., and Moon, I.-C. Augmented variational autoencoders for collaborative filtering with auxiliary information. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (2017), pp. 1139–1148.

[11] Li, S., Kawale, J., and Fu, Y. Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management* (2015), pp. 811–820.

[12] Liang, D. dawenl/vae_cf. *GitHub*.

[13] Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference* (2018), pp. 689–698.

[14] Marlin, B., Zemel, R. S., Roweis, S., and Slaney, M. Collaborative filtering and the missing at random assumption. *arXiv preprint arXiv:1206.5267* (2012).

[15] Mohades, M. M., and Kahaei, M. H. Matrix completion with side information using manifold optimisation. *IET Signal Processing 14*, 2 (2019), 106–114.

[16] Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., and Yang, Q. One-class collaborative filtering. In *2008 Eighth IEEE International Conference on Data Mining* (2008), IEEE, pp. 502–511.

[17] Pilászy, I., Zibriczky, D., and Tikk, D. Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proceedings of the fourth ACM conference on Recommender systems* (2010), pp. 71–78.

[18] Sedhain, S., Menon, A. K., Sanner, S., and Xie, L. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on World Wide Web* (2015), pp. 111–112.

[19] Shani, G., and Gunawardana, A. Evaluating recommendation systems. In *Recommender systems handbook*. Springer, 2011, pp. 257–297.

[20] Strub, F., and Mary, J. Collaborative filtering with stacked denoising autoencoders and sparse inputs. In *NIPS workshop on machine learning for eCommerce* (2015).

[21] Taneja, A., and Arora, A. Recommendation research trends: review, approaches and open issues. *International Journal of Web Engineering and Technology 13*, 2 (2018), 123–186.

[22] Wang, H., Wang, N., and Yeung, D.-Y. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (2015), pp. 1235–1244.

[23] Yelp. Yelp dataset. *Kaggle* (Mar 2020).