Methods 238 (2025) 61–73

Contents lists available at ScienceDirect

Methods

journal homepage: www.elsevier.com/locate/ymeth

Domain alignment method based on masked variational autoencoder for predicting patient anticancer drug response☆

Wei Dai [a,b], Gong Chen [a], Wei Peng [a,b,*], Chuyue Chen [a], Xiaodong Fu [a,b], Li Liu [a,b], Lijun Liu [a,b], Ning Yu [c]

[a] Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650050, China
[b] Computer Technology Application Key Lab of Yunnan Province, Kunming University of Science and Technology, Kunming 650050, China
[c] State University of New York, The College at Brockport, Department of Computing Sciences, 350 New Campus Drive, Brockport, NY 14422, United States

ARTICLE INFO

Keywords:
Patients' response to anticancer drugs
Transfer learning
Domain alignment
Mask predictor

ABSTRACT

Predicting the patient's response to anticancer drugs is essential in personalized treatment plans. However, due to significant distribution differences between cell line data and patient data, models trained well on cell line data may perform poorly on patient anticancer drug response predictions. Some existing methods use transfer learning strategies to implement domain feature alignment between cell lines and patient data and leverage knowledge from cell lines to predict patient anticancer drug responses. This study proposes a domain alignment method based on masked variational autoencoders, MVAEDA, to predict patient anticancer drug responses. The model constructs multiple variational autoencoders (VAEs) and mask predictors to extract specific and domain-invariant features of cell lines and patients. Then, it masks and reconstructs the gene expression matrix, using generative adversarial training to learn domain-invariant features from the cell line and patient domains. These domain-invariant features are then used to train a classifier. Finally, the final trained model predicts the anticancer drug response in the target domain. Our model is experimentally evaluated on the clinical dataset and the preclinical dataset. The results show that our method performs better than other state-of-the-art methods.

1. Introduction

Cancer remains a leading cause of death worldwide despite advancements in diagnosis and treatment. Individual patient responses to anticancer drugs vary significantly due to cancer heterogeneity, highlighting the need for personalized therapy. Hence, accurate drug response prediction can optimize treatment outcomes, minimize side effects, and reduce unnecessary costs. Recently, large-scale drug screening projects like the Cancer Cell Line Encyclopedia (CCLE) [1] and the Genomics of Drug Sensitivity in Cancer (GDSC) [2] have generated valuable cell line-drug response datasets for developing computational models to predict anticancer drug response. Although patient gene expression data are richly recorded. The distribution gap between cell line and patient data hinders the effectiveness of models trained solely on cell lines. Previous works like DeepDSC [3], DeepCDR [4], DIPK [5], VAE-XGBoost [6], MOFGCN [7] and NIHGCN [8] have made significant strides in predicting drug response in cell lines using various approaches. DeepDSC employs autoencoders and Morgan fingerprints, DeepCDR and DIPK combine multi-omics features, VAE-XGBoost uses variational autoencoders and XGBoost, and MOFGCN and NIHGCN utilize heterogeneous graph convolution. Despite their success on cell lines, these models often struggle to generalize to patient data, such as The Cancer Genome Atlas (TCGA).

Transfer learning [9] can effectively solve the above problems. Its core idea is to apply the knowledge learned in the source domain to the target domain for downstream tasks, so that the model can adapt to the environment of the target domain and improve the model performance. When predicting the response of patients to anticancer drugs, transfer learning regards cancer cell line data as the source domain and clinical cancer patient data as the target domain. A variety of transfer learning-based methods for predicting drug response have been proposed, which can be roughly divided into two categories. One is the model-based approach, which consists of two stages: the first stage is to pre-train the model using source domain data, and the second stage is to fine-

☆ This article is part of a special issue entitled: 'Natural language processing - YMETH' published in Methods.

* Corresponding author at: Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming, 650050, China.
E-mail addresses: daiwei@kust.edu.cn (W. Dai), weipeng1980@gmail.com (W. Peng), ieall@kmust.edu.cn (L. Liu), nyu@brockport.edu (N. Yu).

https://doi.org/10.1016/j.ymeth.2025.03.012
Received 19 August 2024; Received in revised form 3 February 2025; Accepted 14 March 2025
Available online 14 March 2025
1046-2023/© 2025 Elsevier Inc. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

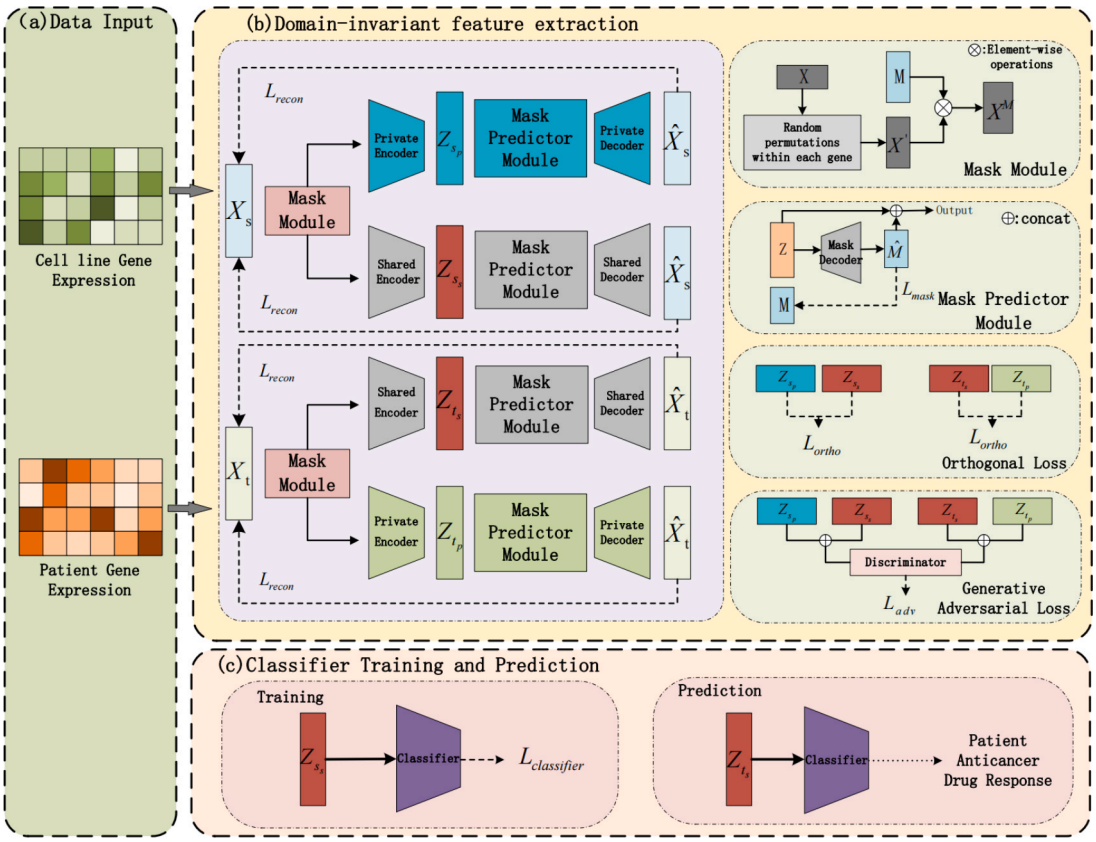


Fig. 1. workflow of MVAEDA. (a) MVAEDA input data. MVAEDA input includes cell line gene expression matrix and patient gene expression matrix. (b) Domain-invariant feature extraction. Mask the data and extract it using a shared encoder. (c) Classifier training and prediction. Use the trained model to predict patient anticancer drug response.

tune the pre-trained model using labeled target domain data. For example, Ma et al. proposed TCPR [10], which is a small sample learning model that is first trained on a large-scale cell line dataset and then fine-tuned using a labeled patient dataset to make it suitable for predicting patients' anticancer drug responses. However, this type of method still requires labeled target domain data. Another class of methods is domain-alignment based methods, which do not require labeled target domain data. It maps the source domain data and the target domain data into the same latent space and trains the model using some constraints, including MMD (maximum mean distribution difference) [11,12], covariance difference [13], and generative adversarial methods [14]. This type of method can not only effectively reduce the difference between the source domain and the target domain, but also achieve feature adaptation between the two domains. For example, Chen and Ma et al. [11] mapped cell line data and clinical sample data into a latent space and used MMD loss to keep the feature distribution of the source domain and the target domain consistent, obtaining effective feature representation to improve model performance. ADAE [15] introduces an adversarial model to remove the confusing information learned from the autoencoder, thereby obtaining more effective information and features in the source and target domains for downstream tasks. He and Wu et al. proposed CODE-AE [12], which constructs three autoencoders and an adversarial module to extract shared and private features of the target and source domains, and then uses the shared features for downstream tasks. Here, the 'private feature' refers to the domain-specific features.

Traditional autoencoders (AEs) compress input data into a low-dimensional latent representation through an encoder, and then reconstruct it back to the original input data through a decoder. However, Traditional autoencoders (AEs) often overlook the feature distribution of the data during encoding. The variational autoencoders (VAE) [16] can solve this problem well. VAE is a generative model that combines autoencoders' encoding ability and the variational inference principle. During the encoding process, the input data is mapped to a probability distribution, and the decoder samples from this distribution to generate data to reconstruct the original input data. Its goal is to minimize the reconstruction error and maximize the similarity between the latent distribution and the prior distribution (achieved through KL divergence). By encoding it into a probability distribution and introducing a regularization term (KL divergence), the model considers the feature distribution of the data, effectively encoding the source domain data into features that approximate the target domain. This approach alleviates distribution differences between the source domain and the target domain during transfer learning.

Masked Language Modeling (MLM) Pretraining is a popular pre-training method in large language models (LLMs). It trains the model by randomly masking some words in the input sequence and using the context to predict the masked content. The low-dimensional embedding obtained in this way can effectively capture the contextual relationship and semantic features, thereby improving the model's understanding and prediction ability of language structure, for example, BERT proposed by Devlin Jacob et al. [17]. Similar strategies are also applied to single-cell clustering. For example, scMAE [18] designs a masked autoencoder model to predict the masked gene expression data. Finally, it effectively learns cell low-dimensional representations and improves the performance of downstream task models.

Based on the above factors and inspirations, we propose a new domain-alignment method based on masked variational autoencoders, MVAEDA, to predict patients' anticancer drug response. It is shown in Fig. 1. First, we construct three variational autoencoders (VAE) and mask predictor modules to extract shared features (domain-invariant features) and private features of the source and target domains, maximize the difference between their shared and private features through

Table 1
TCGA drug sensitivity and resistance sample size.

Drugs	CCLE		TCGA	
	resistant	sensitive	resistant	sensitive
Cisplatin	299	369	20	20
5-fluorouracil	342	325	11	10
Gemcitabine	334	343	23	23
Sorafenib	278	326	13	13
Temozolomid	282	378	23	23

Table 2
PDTC partial drug sensitivity and resistance sample size.

Drugs	**CCLE**		**TCGA**	
	resistant	**sensitive**	**resistant**	**sensitive**
Wee1 Inhibitor	329	335	10	10
GW441756	363	292	10	10
Motesanib	368	285	10	10
Camptothecin	263	324	10	10
Gefitinib	369	297	10	10

the orthogonality constraints between them, and utilize a generative adversarial approach to align the feature distributions between the source and target domains. Finally, we will train a classifier using the domain-invariant features of the source domain and predict patients' anticancer drug response with the trained model. We apply this model for samples from the clinical dataset of The Cancer Genome Atlas (TCGA) and the preclinical data of breast cancer patient-derived xenograft (PDX)-derived tumor cell (PDTC) to predict anticancer drug response. The results show that our method performs excellently compared with other state-of-the-art methods. In summary, compared with previous methods, our contributions are as the follows:

MVAEDA leverages variational autoencoders (VAEs) combined with a novel mask strategy to learn domain-invariant feature representations. A dedicated mask predictor guides the model to learn domain-invariant features by effectively encoding source and target domain data.

MVAEDA constructs three distinct variational autoencoders (VAEs) to capture shared (domain-invariant) features, as well as private features specific to the source and target domains. This multi-VAE architecture facilitates a more robust handling of data uncertainty and promotes a smoother, more structured latent space.

Extensive experiments were conducted to evaluate the effectiveness of the MVAEDA model on a clinical dataset from The Cancer Genome Atlas (TCGA) and a preclinical dataset from Patient-Derived Tumor Cell (PDTC). The results demonstrate that MVAEDA outperforms state-of-the-art algorithms for the majority of tested drugs. Furthermore, integrating features extracted by MVAEDA with DeepDCR and DeepDSC models significantly improves their performance, validating the effectiveness of our feature extraction method.

2. Material

We utilized gene expression profiles from cell lines and tumor tissue samples in a previous research [12], including 1,305 cancer cell line samples from the Cancer Cell Line Encyclopedia (CCLE) database [1] and 9,808 tumor tissue samples from The Cancer Genome Atlas (TCGA) [19]. We focused on the top 1000 genes with the highest unique expressions from cancer cell lines and tumor tissue samples respectively according to [20], and combined the two gene sets to obtain a total of 1426 genes. Therefore, both patients and cell lines only focus on the gene expression values of these 1426 genes. The test datasets were the clinical dataset of The Cancer Genome Atlas (TCGA) and the preclinical data of breast cancer patient-derived xenograft (PDX)-derived tumor cell (PDTC) [21]. To label patient samples in TCGA from [12], we defined sensitivity based on "new tumor event days after treatment.". The TCGA data set includes five drugs: cisplatin (Cis), 5-fluorouracil (Fu), gemcitabine (Gem), sorafenib (Sor) and temozolomide (Tem). For PDTC, we selected 50 drugs with known protein targets and for which cell line response data were available in GDSC. The median AUC value across all drugs in PDTC was used as the classification threshold to construct a binary label for the drug response of the PDTC samples. The number of labeled samples for the five drugs in TCGA and some drugs in PDTC are shown in Table 1 and Table 2, respectively.

3. Method

Our MVAEDA model involves two key steps to predict patients' anticancer drug responses (see Fig. 1):Step 1: Domain-invariant feature extraction; Step 2: Patient drug sensitivity classifier training and prediction. At the first step, three Variational Autoencoders (VAEs) are employed to encode shared and specific features for samples in the source and target domains. Masking and reconstructing techniques are utilized to facilitate this feature learning process. A WGAN-GP architecture is employed to align shared features between the source and target domains.This step involves minimizing three loss functions: reconstruction loss, orthogonality loss, and the generative-adversarial loss of the WGAN-GP. At the second step, the domain-invariant features learned in the first step are utilized as input for a classifier to predict patients' anticancer drug responses.

3.1. Domain-invariant feature extraction

We regard cell line gene expression data $X_s = \{x_s^{(i)}\}_{i=1}^{N_s} \in \mathbb{R}^{N_s \times 1426}$ and patient gene expression data $X_t = \{x_t^{(i)}\}_{i=1}^{N_t} \in \mathbb{R}^{N_t \times 1426}$ as source and target domain data, respectively, where $N_s = 1305$ and $N_t = 9808$ represent the number of cell line and patient data samples used in pre-training. We use these data to pre-train VAE to extract domain-invariant features.

We mask the gene expression matrices to introduce variability and perturbations, thereby obtaining the cell-line-masked gene expression matrix X_s^M and the patient-masked gene expression matrix X_t^M. The progress involves three steps: (1) Gene expression values within each gene are randomly shuffled in both the source and target domains. (2) Use Bernoulli distribution to generate the mask matrix $M = \{M_s, M_t\}$ in the source and target domains, respectively. The mask matrix is the same size as the gene expression matrix, determining which gene expression values are modified. A value of 1 indicates modification, while 0 leaves the value unchanged. (3) Obtain the masked gene expression matrix $X^M = \{X_s^M, X_t^M\}$ by applying element-wise operations. The calculation can be expressed as:

$$X_{ij}^M = X_{ij} \bullet (1 - M_{ij}) + X'_{ij} \bullet M_{ij} \quad (1)$$

where X_{ij} represents the element in the i-th row and j-th column of the original gene expression matrix $X = \{X_s, X_t\}$, X'_{ij} denotes the element in the i-th row and j-th column of the randomly permuted gene expression matrix, X_{ij}^M represents the element in the i-th row and j-th column of the masked gene expression matrix.

Next, we built three variational autoencoders: VAE_{source}, VAE_{target}, and VAE_{shared}, which respectively input the cell line and patient masked gene expression matrix to extract private cell line features Z_{s_p}, the private patient feature Z_{t_p}, the domain-invariant features shared by cell

lines Z_{s_s} and the patients Z_{t_s}. The formula for the variational autoencoders is as follows:

$$\begin{cases} \mu = RELU\left(X^M W_{\mu 1} + b_{\mu 1}\right) W_{\mu 2} + b_{\mu 2} \\ \sigma^2 = RELU\left(X^M W_{\sigma 1} + b_{\sigma 1}\right) W_{\sigma 2} + b_{\sigma 2} \\ Z = \mu + \sigma^2 \odot \epsilon \end{cases} \tag{2}$$

where $W_{\mu 1}, W_{\sigma 1} \in \mathbb{R}^{1426\times 128}$ 和 $W_{\mu 2}, W_{\sigma 2} \in \mathbb{R}^{128\times 32}$, X^M represents X_s^M or X_t^M, μ represents the mean, σ^2 denotes the variance, ∈ represents Gaussian noise, and 32-dimensional features Z represents Z_{s_p} or Z_{t_p} or Z_{s_s} or Z_{t_s}.

We utilize the learned latent features to reconstruct the gene expression matrix. The reconstruction process involves four components: source domain private reconstruction, source domain shared reconstruction, target domain private reconstruction and target domain shared reconstruction. Source domain private reconstruction uses the source domain private features from the source domain mask gene matrix to reconstruct the original gene expression matrix of the source domain. Source domain shared reconstruction uses the shared features from the source domain mask gene matrix to reconstruct the original gene expression matrix of the source domain. Target domain private reconstruction utilizes the target domain private features from the target domain mask gene matrix to reconstruct the original gene expression matrix of the target domain. Target domain shared reconstruction employs the shared features from the target domain mask gene matrix to reconstruct the original gene expression matrix of the target domain.

Take source domain private reconstruction for example, this process involves two steps: (1) The mask predictor uses the latent embedding information Z_{s_p} obtained from VAE_{source} to predict whether the input X_s^M is masked and obtains the predicted mask matrix $\widehat{M_{s_p}}$ The mask predictor is implemented using the following formula:

$$\widehat{M} = F(Z, W) \tag{3}$$

where $F = \{F_{s_p}, F_{t_p}, F_s\}$. is the mask predictor, implemented using the linear layer, and the weights are $W_* = \{W_{s_p}, W_{t_p}, W_s\}$. $Z = \{Z_{s_p}, Z_{t_p}, Z_{s_s}, Z_{t_s}\}$. Therefore, different mask predictors are constructed according to the different inputs Z to generate four mask matrices $\widehat{M} = \{\widehat{M_{s_p}}, \widehat{M_{t_p}}, \widehat{M_{s_s}}, \widehat{M_{t_s}}\}$ of cell line private features, patient private features, cell line domain-invariant features, and patient domain-invariant features. There are two mask predictors for cell line and patient private features respectively, and one mask predictor shared by cell line and patient domain-invariant features. The model uses the cross entropy loss between the predicted mask matrix $\widehat{M}$ and the true mask matrix M to calculate the mask estimation loss.

$$\mathscr{L}_{d_{mask}} = -\sum_{ij} M_d^{(ij)} \log\left(\widehat{M_d^{(ij)}}\right) \tag{4}$$

where $d = \{s_p, s_s, t_p, t_s\}$ represents the private features and shared features from the source domain and the target domain, $M_d^{(ij)}$ and $\widehat{M_d^{(ij)}}$ represents the element at the i-th row and j-th column of the input mask matrix M and predicted mask matrix $\widehat{M}$ of the corresponding domain. Specially, we use the following cross entropy loss to train mask predictor of the source domain private features:

$$\mathscr{L}_{s_p-mask} = -\sum_{ij} M_s^{(ij)} \log\left(\widehat{M_{s_p}^{(ij)}}\right) \tag{5}$$

The predicted mask matrix $\widehat{M}$ and the potential embedding feature Z are concatenated and input into the decoder to reconstruct the original gene expression. The formula is as follows:

$$\widehat{x_d} = \text{Decoder}(\widehat{M_d} \otimes Z_d) \tag{6}$$

where ⊕ represents concatenation. Decoder is a two-layer fully connected decoder. Because there are invariant features and shared features in the source domain and the target domain as well as the corresponding mask prediction matrix, four reconstructed gene expression matrices are output. Then the model calculates the reconstruction loss of these four gene expression matrices respectively to optimize the model.

In the reconstruction loss, we also assign different weights to masked and unmasked genes. Taking the source domain private features as an example, the weighted mean square error (MSE) loss function used for reconstruction is as follows:

$$\mathscr{L}_{s_p-MSE} = \frac{1}{N_s} \sum_{ij} \psi_s^{(ij)} \left| X_s^{(ij)} - \widehat{X_{s_p}^{(ij)}} \right|^2 \tag{7}$$

where $\widehat{X_{s_p}^{(ij)}}$ represents the element of the i-th row and j-th column of the predicted matrix of the source domain private reconstruction, which is calculated by the source domain private feature information Z_{s_p} and the corresponding prediction mask matrix $\widehat{M_{s_p}^{(ij)}}$ (see Formula (8)). The weight $\psi_s^{(ij)}$ is given by Formula 9, which is calculated based on the mask matrix $M_s^{(ij)}$ and the hyperparameter λ (λ = 0.75). The parameter λ determines the importance of the masked samples.

$$\widehat{X_{s_p}^{(i_)}} = \text{Decoder}\left(Z_{s_p}^i \oplus \widehat{M_{s_p}^{(i_)}}\right) \tag{8}$$

$$\psi_s^{(ij)} = M_s^{(ij)} \cdot \lambda + \left(1 - M_s^{(ij)}\right) \cdot (1 - \lambda) \tag{9}$$

Hence, the total loss of the source domain private reconstruction is expressed as:

$$\mathscr{L}_{s_p-recon} = (1-\alpha)\mathscr{L}_{s_p-MSE} + \alpha\mathscr{L}_{s_p-mask} + \beta KL\left(\mu, \sigma^2\right) \tag{10}$$

where $\mathscr{L}_{s_p-MSE}$ and $\mathscr{L}_{s_p-mask}$ represent the mask-weighted mean square error (MSE) loss and mask estimation loss of source domain private reconstruction, respectively. $KL(\mu, \sigma^2) = \frac{1}{2}(-\log\sigma^2 + \mu^2 + \sigma^2 - 1)$, represents the KL divergence of the source domain private autoencoder. α and β are balancing hyperparameters, set to 0.7 and 0.1, respectively.

We use a similar approach to calculate private and shared reconstruction losses in both the source and target domains. Notably, the shared reconstructions in the source and target domains utilize the same encoder, mask predictor and decoder. The total loss for gene expression matrix reconstruction is then determined:

$$\mathscr{L}_{recon} = \mathscr{L}_{s_p-recon} + \mathscr{L}_{s_s-recon} + \mathscr{L}_{t_p-recon} + \mathscr{L}_{t_s-recon} \tag{11}$$

Additionally, an orthogonality loss is adopted to maximize the difference between the extracted private and shared features:

$$\mathscr{L}_{ortho} = \frac{1}{n}\left\{ \left[\left(\frac{Z_{s_p}}{\|Z_{s_p}\|_2} \right)^T \frac{Z_{s_s}}{\|Z_{s_s}\|_2} \right]^2 + \left[\left(\frac{Z_{t_p}}{\|Z_{t_p}\|_2} \right)^T \frac{Z_{t_s}}{\|Z_{t_s}\|_2} \right]^2 \right\} \tag{12}$$

Finally, we adopt the WGAN-GP generative adversarial network architecture [22] to train our model and align the shared features between the source and target domains. This architecture involves a discriminator network (D) that introduces an additional objective constraint equation to encourage domain-invariant features of cell lines and patients generated by the shared autoencoders.

$$\mathscr{L}_{adv}: \begin{cases} \mathscr{L}_{critic} = \frac{1}{N_t}\sum_{ij} D\left(Z_t^{(ij)}\right) - \frac{1}{N_s}\sum_{ij} D\left(Z_s^{(ij)}\right) + \gamma\left(\|\nabla_{\tilde{Z}} D\left(\tilde{Z}\right)\|_2 - 1\right)^2 \\ \mathscr{L}_{gen} = -\frac{1}{N_t}\sum_{ij} D\left(Z_t^{(ij)}\right) \end{cases} \tag{13}$$

where $Z_t^{(ij)} = Z_{t_s}^{(ij)} \oplus Z_{t_p}^{(ij)}$ and $Z_s^{(ij)} = Z_{s_s}^{(ij)} \oplus Z_{s_p}^{(ij)}$, $\|\cdot\|$ denotes the L2 norm, γ is set to 10, and $\tilde{Z} = \eta Z_s + (1-\eta) Z_t$, where η is sampled from the standard normal distribution $U(0,1)$. Summary, Algorithm lists the

Table 3
Comparison evaluation metric values of TCGA drug.

Metric	Algorithm	Cis	Sor	Tem	Gem	Fu	Mean
AUC	Codeae_adv	0.60400	0.55148	**0.72817**	0.52968	0.63273	0.60921
	Codeae_mmd	0.56150	0.58935	0.70170	0.49603	0.69091	0.60790
	ADAE	**0.76850**	0.59468	0.69414	0.52590	0.60636	0.63792
	DAE	0.55350	0.51598	0.68960	0.47335	0.60909	0.56830
	DSN-MMD	0.64300	0.49704	0.70208	0.49187	0.62546	0.59189
	DSN-DANN	0.56600	0.65917	**0.72817**	0.48355	0.63818	0.61501
	DeepCDR	0.49563	0.45888	0.61028	0.44303	0.48727	0.49902
	DeepCDR + F	0.56893	0.65568	0.59136	0.49151	0.55973	0.57344
	DeepDSC	0.53802	0.47296	0.58287	0.50136	0.45973	0.51099
	DeepDSC + F	0.54521	0.69506	0.61858	0.53322	0.60273	0.59896
	SCAD	0.48300	0.38364	0.46314	0.39290	0.62609	0.46975
	scDEAL	0.37200	0.42249	0.65785	0.38110	0.52364	0.47142
	MVAEDA	0.68945	**0.72379**	0.70210	**0.60491**	**0.71509**	**0.68707**
AUPRC	Codeae_adv	0.63006	0.61815	**0.76191**	0.53082	0.66825	0.64184
	Codeae_mmd	0.54004	0.62229	0.70386	0.51939	0.65570	0.60826
	ADAE	**0.73151**	0.61657	0.68886	0.52540	0.55945	0.62436
	DAE	0.59010	0.57810	0.71073	0.46473	0.63400	0.59553
	DSN-MMD	0.64836	0.60163	0.73793	0.49206	0.53634	0.60326
	DSN-DANN	0.56590	0.68058	0.75656	0.51690	0.56250	0.61649
	DeepCDR	0.55191	0.55741	0.63662	0.48177	0.52085	0.54971
	DeepCDR + F	0.60517	0.71529	0.61967	0.53357	0.58605	0.61195
	DeepDSC	0.57275	0.55600	0.62571	0.51837	0.49988	0.55454
	DeepDSC + F	0.59464	0.74287	0.64852	0.56267	0.60318	0.63038
	SCAD	0.49247	0.43659	0.49675	0.48422	0.58280	0.49857
	scDEAL	0.47414	0.56389	0.65100	0.43555	0.57873	0.54066
	MVAEDA	0.71989	**0.75527**	0.71416	**0.60987**	**0.73035**	**0.70591**
F1	Codeae_adv	0.20971	0.55556	0.65437	0.41284	0.35148	0.43679
	Codeae_mmd	0.68410	0.63196	**0.68726**	0.24866	0.48667	0.54773
	ADAE	0.34113	0.40000	0.65490	0.14247	0.03333	0.31437
	DAE	0.65670	**0.63461**	0.66667	0.28295	0.48465	0.54512
	DSN-MMD	**0.68770**	0.50571	0.65273	0.32953	0.54643	0.54442
	DSN-DANN	0.66737	0.59415	0.66862	0.35753	0.61283	**0.58010**
	DeepCDR	0.48317	0.53307	0.64143	0.40885	0.41913	0.49713
	DeepCDR + F	0.56789	0.58454	0.61450	0.39318	0.50544	0.53311
	DeepDSC	0.40410	0.43158	0.50694	0.26102	0.28574	0.37788
	DeepDSC + F	0.43159	0.48773	0.48849	0.37644	0.33604	0.42406
	SCAD	0.59604	0.27812	0.26019	**0.49579**	**0.66886**	0.45980
	scDEAL	0.34777	0.55556	0.65484	0.10106	0.06970	0.34579
	MVAEDA	0.47924	0.55155	0.61436	0.17278	0.38334	0.44025
ACC	Codeae_adv	0.53000	0.38462	0.50870	**0.52174**	0.55238	0.49949
	Codeae_mmd	0.57000	0.52308	0.56957	0.46957	**0.61905**	0.55025
	ADAE	0.52000	0.50000	0.48696	0.48261	0.52381	0.50268
	DAE	0.53000	0.46923	0.50000	0.46522	0.51429	0.49575
	DSN-MMD	**0.63500**	0.35385	0.49130	0.44348	0.60952	0.50663
	DSN-DANN	0.57500	0.49231	0.50435	0.49130	0.56191	0.52497
	DeepCDR	0.49550	0.44231	0.52478	0.46804	0.49524	0.48517
	DeepCDR + F	0.50550	0.51923	0.50391	0.49478	0.52000	0.50868
	DeepDSC	0.52075	0.49423	0.55609	0.49370	0.49429	0.51181
	DeepDSC + F	0.49950	0.50385	0.50848	0.50283	0.50286	0.50350
	SCAD	0.48500	0.45714	0.45652	0.33077	0.51304	0.44849
	scDEAL	0.46000	0.38462	0.49565	0.46087	0.53333	0.46689
	MVAEDA	0.59250	**0.63077**	**0.64217**	0.51174	0.61619	**0.59867**
MCC	Codeae_adv	0.13942	−0.36116	−0.00769	0.03033	0.12452	−0.01491
	Codeae_mmd	0.19781	0.04497	0.18307	−0.08223	**0.24493**	0.11771
	ADAE	0.10993	0.00000	−0.08940	−0.06090	0.00309	−0.00746
	DAE	0.08091	−0.13774	0.00000	−0.11523	0.03420	−0.02757
	DSN-MMD	**0.28530**	−0.37870	−0.05963	−0.12631	0.23713	−0.00844
	DSN-DANN	0.17612	−0.10145	0.02981	−0.02435	0.14881	0.04579
	DeepCDR	0.00021	−0.14266	0.05990	−0.07133	−0.01285	−0.03335
	DeepCDR + F	0.01475	0.04679	0.01206	−0.01780	0.06266	0.02369
	DeepDSC	0.06050	−0.01700	0.12877	−0.03613	−0.04088	0.01905
	DeepDSC + F	0.00324	0.01038	0.01991	0.01101	0.00682	0.01027
	SCAD	−0.04128	−0.11603	−0.10171	−0.44993	0.03584	−0.13462
	scDEAL	−0.09035	−0.36116	−0.04202	−0.13156	0.05000	−0.11502
	MVAEDA	0.22814	**0.29291**	**0.29498**	**0.03906**	0.24172	**0.21936**
Recall	Codeae_adv	0.16000	0.76923	0.93044	0.38261	0.36000	0.52046
	Codeae_mmd	**0.93000**	0.81539	0.93913	0.18261	0.38000	0.64943
	ADAE	0.44000	0.60000	0.97391	0.08696	0.02000	0.42417
	DAE	0.90000	0.92308	**1.00000**	0.29565	0.52000	0.72775
	DSN-MMD	0.80000	0.66154	0.95652	0.28696	0.66000	0.67300
	DSN-DANN	0.85000	0.80000	**1.00000**	0.30435	**0.74000**	0.73887
	DeepCDR	0.65250	0.66692	0.8600	0.43696	0.47300	0.61788
	DeepCDR + F	0.81750	0.81615	0.90044	0.54739	0.66400	**0.74910**
	DeepDSC	0.46200	0.46615	0.52087	0.22826	0.29600	0.39466

(*continued on next page*)

Table 3 (*continued*)

Metric	Algorithm	Cis	Sor	Tem	Gem	Fu	Mean
	DeepDSC + F	0.63450	0.72308	0.70478	0.54913	0.51400	0.62510
	SCAD	0.76000	0.22000	0.19130	**0.66154**	0.98261	0.56309
	scDEAL	0.35000	0.76923	0.95652	0.06087	0.04000	0.43532
	MVAEDA	0.46400	0.50462	0.59565	0.17913	0.32600	0.41388
Precision	Codeae_adv	**0.85526**	0.43478	0.50587	0.48374	0.75585	0.60710
	Codeae_mmd	0.54161	0.51796	0.54375	0.40762	0.68000	0.53819
	ADAE	0.80000	0.70000	0.49333	0.40333	**0.90000**	0.65933
	DAE	0.51801	0.48367	0.50000	0.39540	0.51191	0.48180
	DSN-MMD	0.61204	0.40975	0.49556	0.39813	0.67240	0.51758
	DSN-DANN	0.55039	0.54400	0.50222	0.46764	0.55810	0.52447
	DeepCDR	0.57639	0.46841	0.51872	0.46152	0.50131	0.50527
	DeepCDR + F	0.54654	0.57500	0.51608	0.62004	0.57461	0.56645
	DeepDSC	0.64912	0.55248	0.60670	0.44282	0.55843	0.56191
	DeepDSC + F	0.66735	0.63898	0.62433	0.70041	0.73126	0.67247
	SCAD	0.49137	0.38952	0.40965	0.39678	0.50708	0.43888
	scDEAL	0.42540	0.43478	0.49788	0.30000	**0.90000**	0.51161
	MVAEDA	0.72714	**0.74439**	**0.66958**	**0.74587**	0.81112	**0.73962**

pseudocode of domain-invariant feature extraction.

Algorithm Domain-Invariant Feature Extraction
Input: Source Domain X_s, Target Domain X_t, batch size b, pretrain epoch e_p, GAN epoch e_g, number of steps per encoders update n_g
Output: Source domain Shared Feature Z_{s_s} and Target domain shared Feature Z_{t_s}
for epoch = 1 **to** e_p **do**
  for t = 1 **to** $\frac{\min(N_s, N_t)}{b}$ **do**
    sample $\{x_s\}$, $\{x_t\}$ of size b from X_s, X_t
    Mask samples $\{x_s\}$, $\{x_t\}$ to get the mask samples $\{x_s^M\}$, $\{x_t^M\}$
    Calculate the loss $\mathscr{L}_{recon}$, $\mathscr{L}_{ortho}$
Update VAE_{source}, VAE_{target}, VAE_{shared}, $Decoder_{mask}$ with $\mathscr{L}_{recon} + \mathscr{L}_{ortho}$
  end for
end for
for epoch = 1 **to** e_g **do**
  for t = 1 **to** $\frac{\min(N_s, N_t)}{b}$ **do**
    sample $\{x_s\}$, $\{x_t\}$ of size b from X_s, X_t
    Mask samples $\{x_s\}$, $\{x_t\}$ to get the mask samples $\{x_s^M\}$, $\{x_t^M\}$
    Calculate the loss $\mathscr{L}_{critic}$ to update D
    if t%n_g == 0 **then**
      Calculate the loss $\mathscr{L}_{recon}$, $\mathscr{L}_{ortho}$, $\mathscr{L}_{gen}$
      Update VAE_{source}, VAE_{target}, VAE_{shared}, $Decoder_{mask}$ with $\mathscr{L}_{recon} + \mathscr{L}_{ortho} + \mathscr{L}_{gen}$
    end if
  end for
end for

3.2. Patient drug sensitivity classifier training

In this stage, we use the domain-invariant features of the source domain to train a classifier C to predict the patient's anticancer drug response. The classifier C consists of two linear layers, a batch normalization layer, a RELU activation function layer and a sigmoid layer.

The training loss is as follows:

$$\mathscr{L}_{classifier} = -\frac{1}{N_{s-train}} \sum_{i=1}^{N_{s-train}} y_i \log\sigma\left(C\left(Z_{s_s}^{(i)}\right)\right) + (1-y_i)\log\left[1-\sigma\left(C\left(Z_{s_s}^{(i)}\right)\right)\right] \tag{14}$$

Where $Z_{s_s}^{(i)}$ represents the i-th sample in Z_{s_s} (domain-invariant features of the source domain), $N_{s-train}$ is the number of training samples of the source domain data, $\sigma(\cdot)$ is the sigmoid function, and y_i represents the true label corresponding to the source domain data. Finally, we input the domain-invariant features of the patients into classifier C and use the trained classifier C to predict the patients' anticancer drug response.

4. Experiment

4.1. Baseline and Experimental Setup

To access the performance of our model in predicting patients' anticancer drug responses, we conduct a comprehensive comparison with state-of-the-art transfer learning-based methods. These methods include Codeae-adv [12], Codeae-mmd [12], ADAE [15], DAE [23], DSN-MMD [24], DSN-DANN [24], SCAD [25] and scDEAL [11]. In addition, we compare our method with recent anticancer drug response prediction methods on cell line datasets, such as DeepCDR [4] and DeepDSC [3]. Since the two methods lack a cross-domain feature fitting training process, we used the cell line and patient domain-invariant features learned by the model as input of the model to comprehensively evaluate the effectiveness of the learned domain-invariant features. The two modified input feature models are denoted as DeepCDR + F and DeepDSC + F, respectively.

We employed a rigorous evaluation process to train MVAEDA and assess its performance. Our MVAEDA model involves two key steps to predict patients' anticancer drug responses (see Fig. 1): Step 1: Domain-invariant feature extraction; Step 2: Patient drug sensitivity classifier training and prediction. In the first step, we train a model to encode domain-invarient features from source and target domain data. We utilized 1305 cancer cell line samples from the CCLE as the source domain training set and 9808 tumor tissue samples from TCGA as the target domain training set. To generate validation sets, we first filtered the CCLE data, removing disease categories with fewer than two samples. Stratified sampling by disease category was then performed on both the filtered CCLE data and the TCGA data. 10 % of the CCLE samples (134 cell lines) were allocated to the source domain validation set, and an equal number of TCGA samples, stratified by disease category, formed the target domain validation set. Model training employed an early stopping strategy, terminating training if the validation loss did not decrease for 50 epochs. In the second step, we train classifier for each drug under five fold cross-validation. The recorrsponding cell-line response data and the domain-invarient features learned in the first step serve as input. Patient data from the target domain is test set. For each drug, the cell-line response data was divided into five subsets, with four subsets for training the classifier and one subset for validation. The classifier with the highest performance in the validation set was selected to test patient response prediction. The entire process was repeated ten times and the final results were obtained by averaging the evaluation metrics across these ten independent runs of five-fold cross-validation. Seven common evalation metrics, AUC, AUPRC F1, MCC, ACC, Precision, and Recall, were used to evaluation the performance of each method.

For hyperparameter tuning, during domain-invariant feature

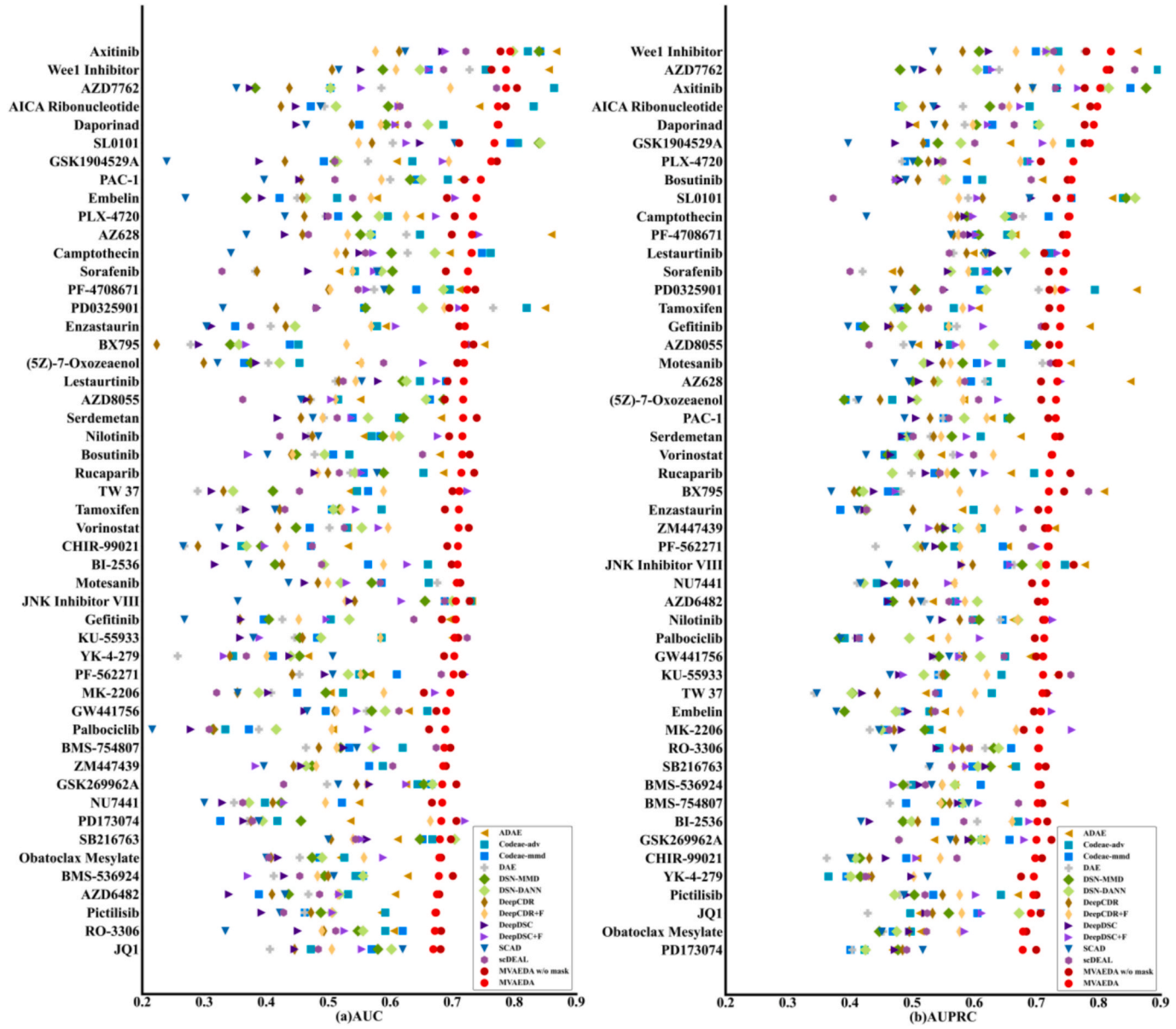


Fig. 2. Comparison of the prediction performance of each method on the PDTC dataset. Comparison of the predicted AUC (a) and AUPRC (b) values for the response prediction of each method for each drug in the PDTC 50 drug dataset.

extraction, we used a cosine annealing learning rate of 0.001 and early stopping with a patience of 50 and 20 epochs for models with and without generative adversarial training, respectively. And grid search was performed on the training epochs to store models at different pre-training epochs. The training epochs without adversarial training were set as (0, 100, 300), while the training epochs with adversarial training were set as (100, 200, 300, 500, 750, 1000, 1500, 2000, 2500, 3000). For the classifier training phase, we set the same training epochs as those used during the adversarial training in the domain-invariant feature extraction step. Early stopping with a patience of 20 epochs was implemented to prevent overfitting. We utilized grid search and cosine annealing to determine optimal learning rates from {0.01 and 0.001}. Parameters for other transfer learning-based methods were set according to recommendations from Codeae [10], SCAD [25] and scDEAL [11], ensuring consistency with previous research. For the anticancer drug prediction method on the cell line dataset, the parameters were set according to those mentioned in the original article.

4.2. Experimental results

4.2.1. Drug response prediction based on clinical TCGA dataset

We evaluated MVAEDA's performance on the TCGA dataset for five selected drugs. Table 3 presents the AUC, AUPRC, F1, MCC, ACC, Precision, and Recall values for each method. MVAEDA achieved the highest AUC and AUPRC on Sor (Sorafenib), Gem (Gemcitabine), and Fu (Fluorouracil). Although its performance on Cis (Cisplatin) was slightly lower, MVAEDA still demonstrated robust results. The highest ACC was achieved on Sor and Tem (Temozolomide), while the highest MCC and Precision were obtained on Sor, Tem, and Gem. Notably, MVAEDA outperformed other methods in the average AUC, AUPRC, MCC, ACC, and Precision across all five drugs. Compared to the next best approach for each average metric, MVAEDA achieved improvements of 4.915 %, 6.407 %, 10.165 %, 4.842 %, and 6.715 %, respectively, demonstrating the model's superior ability to distinguish samples and predict positive classes accurately. However, due to the choice of the classification threshold (set to 0.5 in this study), the model's predictions for the positive class were relatively conservative, leading to slightly lower Recall

Table 4
Average evaluation metric of each method in PDTC dataset.

Method	Mean AUC	AUC Rank	Mean AUPRC	AUPRC Rank	Mean F1	F1 Rank	Mean ACC	ACC Rank	Mean MCC	MCC Rank	Mean Precision	Precision Rank	Mean Recall	Recall Rank
Code-adv	0.59729	5.32	0.60630	6.56	0.46205	4.66	0.54263	5.54	0.09075	5.78	0.60026	7.48	0.47389	5.56
Code-mmd	0.54215	7.46	0.55758	8.68	0.44983	5.16	0.52143	6.34	0.04616	6.84	0.55037	8.98	0.47546	5.78
ADAE	0.64276	4.2	0.64369	5.2	0.37910	7.12	0.53358	5.7	0.07701	5.9	0.70613	5.18	0.44357	7.06
DAE	0.49185	9.8	0.53002	10.36	0.35386	8.28	0.47778	9.36	−0.04550	9.58	0.56863	8.94	0.39167	8.22
DSN-MMD	0.52764	7.88	0.54011	9.3	**0.46338**	**4.76**	0.51180	7.44	0.03214	7.42	0.54675	9.4	**0.49509**	**5.26**
DSN-DANN	0.54686	6.88	0.55789	8.4	0.43617	5.2	0.51895	6.36	0.04720	6.7	0.58255	8.2	0.45464	6.1
DeepCDR	0.44357	11.72	0.52684	9.9	0.34229	9.1	0.47006	10.88	−0.06560	10.92	0.55237	9.22	0.40662	7.98
DeepCDR + F	0.54547	8.1	0.60402	6.88	0.37116	7.6	0.51040	7.26	0.03031	7.18	0.67680	5.78	0.49885	5.6
DeepDSC	0.43800	11.94	0.52092	10.5	0.28073	10.94	0.46861	10.92	−0.07590	11.1	0.52694	10.1	0.27566	11.08
DeepDSC + F	0.57494	6.56	0.62736	5.74	0.32271	9.26	0.50204	7.98	0.00920	7.92	0.73517	4.18	0.47113	6.06
SCAD	0.41661	11.88	0.49720	11.56	0.42048	6.4	0.42947	11.52	−0.14359	11.62	0.41923	12.82	0.43530	6.84
scDEAL	0.51596	8.82	0.58387	7.4	0.39979	6.72	0.50588	7.82	0.01892	7.7	0.55923	9.18	0.45834	6.88
MVAEDA w/o mask	0.70974	2.2	0.72441	2.42	0.32369	9.68	**0.57617**	**2.94**	**0.17353**	**3.04**	0.77795	2.98	0.28154	10.72
MVAEDA	**0.71519**	**2.16**	**0.72968**	**2.1**	0.29811	10.04	0.57012	3.22	0.16144	3.2	**0.80979**	**2.46**	0.27169	10.72

Table 5
Results of ablation experiments.

Metric	Algorithm	Cis	Sor	Tem	Gem	Fu	Mean
AUC	**MVAEDA**	0.68945	0.72379	0.70210	0.60491	0.71509	**0.68707**
	MVAEDA w/o mask	0.65745	0.66923	0.70707	0.62458	0.70927	0.67352
	Codeae_adv	0.60400	0.55148	0.72817	0.52968	0.63273	0.60921
	MVAEDA w/o adv	0.71975	0.66544	0.67229	0.62514	0.67345	0.67122
	MVAEDA w/o ortho	0.68765	0.66746	0.70060	0.65501	0.67691	0.67753
AUPRC	**MVAEDA**	0.71989	0.75527	0.71416	0.60987	0.73035	**0.70591**
	MVAEDA w/o mask	0.65781	0.67663	0.70730	0.61872	0.73372	0.67884
	Codeae_adv	0.63006	0.61815	0.76191	0.53082	0.66825	0.64184
	MVAEDA w/o adv	0.74207	0.67189	0.63986	0.62203	0.68779	0.67273
	MVAEDA w/o ortho	0.70036	0.71473	0.69888	0.64970	0.69439	0.69161
F1	**MVAEDA**	0.47924	0.55155	0.61436	0.17278	0.38334	**0.44025**
	MVAEDA w/o mask	0.46313	0.41957	0.60530	0.20427	0.43419	0.42529
	Codeae_adv	0.20971	0.55556	0.65437	0.41284	0.35148	0.43679
	MVAEDA w/o adv	0.29055	0.20944	0.22786	0.16337	0.26044	0.23033
	MVAEDA w/o ortho	0.49591	0.47616	0.61197	0.22795	0.31271	0.42494
ACC	**MVAEDA**	0.59250	0.63077	0.64217	0.51174	0.61619	**0.59867**
	MVAEDA w/o mask	0.58450	0.57923	0.63826	0.52261	0.62762	0.59044
	Codeae_adv	0.53000	0.38462	0.50870	0.52174	0.55238	0.49949
	MVAEDA w/o adv	0.57350	0.54000	0.54435	0.51652	0.55238	0.54535
	MVAEDA w/o ortho	0.61250	0.61154	0.63522	0.53348	0.58571	0.59569
MCC	**MVAEDA**	0.22814	0.29291	0.29498	0.03906	0.24172	**0.21936**
	MVAEDA w/o mask	0.18215	0.17425	0.28405	0.05885	0.29074	0.19801
	Codeae_adv	0.13942	−0.36116	−0.00769	0.03033	0.12452	−0.01491
	MVAEDA w/o adv	0.18024	0.08553	0.11311	0.04120	0.10457	0.10493
	MVAEDA w/o ortho	0.24971	0.26365	0.27784	0.08795	0.19053	0.21394
Recall	**MVAEDA**	0.46400	0.50462	0.59565	0.17913	0.32600	0.41388
	MVAEDA w/o mask	0.43700	0.36615	0.57739	0.18000	0.35800	0.38371
	Codeae_adv	0.16000	0.76923	0.93044	0.38261	0.36000	**0.52046**
	MVAEDA w/o adv	0.22400	0.22308	0.23652	0.16522	0.26000	0.22176
	MVAEDA w/o ortho	0.41900	0.39231	0.59130	0.17652	0.24400	0.36463
Precision	**MVAEDA**	0.72714	0.74439	0.66958	0.74587	0.81112	**0.73962**
	MVAEDA w/o mask	0.64759	0.66471	0.66224	0.72455	0.80112	0.70004
	Codeae_adv	0.85526	0.43478	0.50587	0.48374	0.75585	0.60710
	MVAEDA w/o adv	0.85996	0.83575	0.84476	0.81651	0.77731	0.82686
	MVAEDA w/o ortho	0.71834	0.74127	0.65638	0.71962	0.82231	0.73159

and F1 scores. Nevertheless, MVAEDA's higher MCC further validates its excellent overall performance in predicting both positive and negative classes. Additionally, the domain-invariant features extracted by MVAEDA enhanced the performance of DeepCDR and DeepDSC, demonstrating the effectiveness of our method in capturing domain-invariant features.

4.2.2. Prediction of drug response from preclinical PDTC data

We assessed MVAEDA's performance on the PDTC dataset, predicting responses to 50 drugs. Each drug was treated as a separate task, following the approach in Codeae [10]. Fig. 2 illustrates the individual drug performance, while Table 4 shows the average AUC, AUPRC, F1, MCC, ACC, Precision, and Recall values for all drugs in the PDTC dataset. Compared with other methods, MVAEDA achieved the highest average AUC, AUPRC, MCC, ACC, and Precision. Similar to the TCGA dataset results, the classification threshold (set to 0.5) led to relatively conservative predictions for the positive class, resulting in lower Recall and F1 scores. Despite this, MVAEDA's higher MCC further validates its robust overall prediction performance across positive and negative samples. Furthermore, replacing the raw gene expression profiles with the

Table 6
Results of Different Masking Rates.

Metric	Parameter	Cis	Sor	Tem	Gem	Fu	Mean
AUC	**Mask rate = 40 %**	0.68945	0.72379	0.70210	0.60491	0.71509	**0.68707**
	Mask rate = 20 %	0.68500	0.72521	0.67898	0.60892	0.71309	0.68224
	Mask rate = 30 %	0.68395	0.72024	0.68548	0.60662	0.71891	0.68304
	Mask rate = 50 %	0.67493	0.69432	0.70824	0.60129	0.70491	0.67674
AUPRC	**Mask rate = 40 %**	0.71989	0.75527	0.71416	0.60987	0.73035	**0.70591**
	Mask rate = 20 %	0.69775	0.75445	0.68046	0.59627	0.70621	0.68703
	Mask rate = 30 %	0.69894	0.75889	0.68293	0.60650	0.72583	0.69462
	Mask rate = 50 %	0.69174	0.73118	0.71331	0.59835	0.70127	0.68717
F1	**Mask rate = 40 %**	0.47924	0.55155	0.61436	0.17278	0.38334	**0.44025**
	Mask rate = 20 %	0.41273	0.51212	0.59345	0.23534	0.36029	0.42279
	Mask rate = 30 %	0.41903	0.60498	0.57945	0.20686	0.28417	0.41890
	Mask rate = 50 %	0.47302	0.53707	0.60215	0.22897	0.35908	0.44006
ACC	**Mask rate = 40 %**	0.59250	0.63077	0.64217	0.51174	0.61619	**0.59867**
	Mask rate = 20 %	0.57000	0.62692	0.62391	0.52609	0.59905	0.58919
	Mask rate = 30 %	0.57450	0.66000	0.62217	0.52174	0.58095	0.59187
	Mask rate = 50 %	0.60300	0.62462	0.63435	0.50957	0.59524	**0.59336**
MCC	**Mask rate = 40 %**	0.22814	0.29291	0.29498	0.03906	0.24172	0.21936
	Mask rate = 20 %	0.17761	0.28410	0.25358	0.06024	0.20072	0.19525
	Mask rate = 30 %	0.19638	0.34281	0.25689	0.05712	0.17761	0.20616
	Mask rate = 50 %	0.23316	0.28106	0.28363	0.03492	0.19222	0.20500
Recall	**Mask rate = 40 %**	0.46400	0.50462	0.59565	0.17913	0.32600	0.41388
	Mask rate = 20 %	0.39400	0.43846	0.56174	0.24609	0.30800	0.38966
	Mask rate = 30 %	0.38100	0.55077	0.56087	0.19044	0.23800	0.38422
	Mask rate = 50 %	0.42700	0.48462	0.58000	0.24174	0.34000	**0.41467**
Precision	**Mask rate = 40 %**	0.72714	0.74439	0.66958	0.74587	0.81112	**0.73962**
	Mask rate = 20 %	0.72278	0.74248	0.64527	0.69082	0.72494	0.70526
	Mask rate = 30 %	0.72821	0.74819	0.66144	0.69948	0.82486	0.73244
	Mask rate = 50 %	0.69439	0.72380	0.66806	0.64826	0.79290	0.70548

Table 7
Experimental Results for Different α Parameters.

Metric	Parameter	Cis	Sor	Tem	Gem	Fu	Mean
AUC	$\alpha = \mathbf{0.7}$	0.68945	0.72379	0.70210	0.60491	0.71509	**0.68707**
	α = 0.5	0.66005	0.71752	0.70068	0.60730	0.68564	0.67424
	α = 0.6	0.65125	0.72675	0.69051	0.60448	0.71018	0.67663
	α = 0.8	0.6838	0.71124	0.70034	0.60739	0.71018	0.68259
AUPRC	$\alpha = \mathbf{0.7}$	0.71989	0.75527	0.71416	0.60987	0.73035	**0.70591**
	α = 0.5	0.67687	0.74137	0.70783	0.60545	0.68098	0.68250
	α = 0.6	0.66579	0.73866	0.68518	0.60006	0.72555	0.68305
	α = 0.8	0.71491	0.74248	0.70721	0.60808	0.71776	0.69809
F1	$\alpha = \mathbf{0.7}$	0.47924	0.55155	0.61436	0.17278	0.38334	**0.44025**
	α = 0.5	0.43220	0.54045	0.59369	0.23234	0.29015	0.41777
	α = 0.6	0.42773	0.50704	0.54507	0.21903	0.40605	0.42098
	α = 0.8	0.48556	0.57061	0.61536	0.13688	0.33873	0.42943
ACC	$\alpha = \mathbf{0.7}$	0.59250	0.63077	0.64217	0.51174	0.61619	**0.59867**
	α = 0.5	0.59000	0.62692	0.63435	0.51391	0.57524	0.58808
	α = 0.6	0.57750	0.62077	0.62130	0.51783	0.60095	0.58767
	α = 0.8	0.58750	0.64154	0.62087	0.51044	0.57905	0.58788
MCC	$\alpha = \mathbf{0.7}$	0.22814	0.29291	0.29498	0.03906	0.24172	**0.21936**
	α = 0.5	0.21310	0.26800	0.27928	0.04487	0.15371	0.19179
	α = 0.6	0.17654	0.27080	0.25574	0.04497	0.23414	0.19644
	α = 0.8	0.20282	0.31054	0.25413	0.02966	0.17193	0.19382
Recall	$\alpha = \mathbf{0.7}$	0.46400	0.50462	0.59565	0.17913	0.32600	**0.41388**
	α = 0.5	0.36000	0.50769	0.56522	0.25217	0.25200	0.38742
	α = 0.6	0.39500	0.44615	0.52261	0.23044	0.39400	0.39764
	α = 0.8	0.46300	0.49231	0.63304	0.10261	0.33400	0.40499
Precision	$\alpha = \mathbf{0.7}$	0.72714	0.74439	0.66958	0.74587	0.81112	**0.73962**
	α = 0.5	0.68307	0.73230	0.67638	0.69527	0.76050	0.70950
	α = 0.6	0.68404	0.71728	0.66623	0.63018	0.79636	0.69882
	α = 0.8	0.68900	0.73294	0.63442	0.69355	0.78648	0.70728

domain-invariant features extracted by MVAEDA significantly improved the performance of DeepCDR and DeepDSC. The use of our masking strategy also enhanced the performance, with average AUC, AUPRC, and Precision improving by 0.545 %, 0.527 %, and 3.184 %, respectively, compared to MVAEDA without masking. These results demonstrate the exceptional capability of MVAEDA in predicting patient responses to anticancer drugs in the PDTC dataset.

4.2.3. Ablation Experiment results

To assess the impact of our masking strategy, VAE, adversarial training and orthogonal loss, we conducted ablation experiments on the TCGA dataset. The results in Table 5 indicate that all these components positively contribute to the performance. Removing the masking strategy (MVAEDA w/o mask) resulted in a decrease in all average metrics compared to the complete model. Specifically, the average AUC decreased by 1.355 %, the average AUPRC by 2.707 %, the average F1 by 1.496 %, the average ACC by 0.823 %, the average MCC by 2.135 %,

Table 8
Experimental Results for Different λ Parameters.

Metric	Parameter	Cis	Sor	Tem	Gem	Fu	Mean
AUC	$\lambda = \mathbf{0.75}$	0.68945	0.72379	0.70210	0.60491	0.71509	**0.68707**
	$\lambda = 0.65$	0.68945	0.71260	0.70098	0.60979	0.70400	0.68337
	$\lambda = 0.70$	0.66115	0.71065	0.70106	0.60000	0.72127	0.67883
	$\lambda = 0.8$	0.68640	0.72379	0.69565	0.61021	0.69982	0.68317
AUPRC	$\lambda = \mathbf{0.75}$	0.71989	0.75527	0.71416	0.60987	0.73035	**0.70591**
	$\lambda = 0.65$	0.69264	0.73256	0.69904	0.59351	0.72690	0.68893
	$\lambda = 0.70$	0.67213	0.75777	0.70407	0.59296	0.72263	0.68991
	$\lambda = 0.8$	0.70010	0.75562	0.71324	0.61172	0.69049	0.69423
F1	$\lambda = \mathbf{0.75}$	0.47924	0.55155	0.61436	0.17278	0.38334	**0.44025**
	$\lambda = 0.65$	0.46369	0.46537	0.56997	0.23530	0.28163	0.40319
	$\lambda = 0.70$	0.39162	0.56102	0.63216	0.16534	0.30707	0.41144
	$\lambda = 0.8$	0.41710	0.52762	0.61090	0.25949	0.33259	0.42954
ACC	$\lambda = \mathbf{0.75}$	0.59250	0.63077	0.64217	0.51174	0.61619	**0.59867**
	$\lambda = 0.65$	0.60600	0.60000	0.62391	0.52087	0.58095	0.58635
	$\lambda = 0.70$	0.57100	0.63769	0.64522	0.51087	0.57905	0.58877
	$\lambda = 0.8$	0.59100	0.62308	0.63609	0.52957	0.58571	0.59309
MCC	$\lambda = \mathbf{0.75}$	0.22814	0.29291	0.29498	0.03906	0.24172	**0.21936**
	$\lambda = 0.65$	0.23562	0.23066	0.26276	0.04724	0.17235	0.18973
	$\lambda = 0.70$	0.17148	0.29624	0.29953	0.02716	0.18789	0.19646
	$\lambda = 0.8$	0.22584	0.26617	0.28529	0.06781	0.18170	0.20536
Recall	$\lambda = \mathbf{0.75}$	0.46400	0.50462	0.59565	0.17913	0.32600	**0.41388**
	$\lambda = 0.65$	0.41200	0.40923	0.55304	0.24435	0.23200	0.37012
	$\lambda = 0.70$	0.34600	0.51538	0.63130	0.17043	0.27200	0.38702
	$\lambda = 0.8$	0.33300	0.48154	0.59304	0.25304	0.29600	0.39133
Precision	$\lambda = \mathbf{0.75}$	0.72714	0.74439	0.66958	0.74587	0.81112	**0.73962**
	$\lambda = 0.65$	0.69611	0.70967	0.67320	0.61587	0.83953	0.70688
	$\lambda = 0.70$	0.68226	0.73772	0.66328	0.70368	0.82436	0.72226
	$\lambda = 0.8$	0.74579	0.72337	0.66282	0.69656	0.78605	0.72292

Table 9
Experimental Results for Different Learning Rates.

Metric	Lr	Cis	Sor	Tem	Gem	Fu	Mean
AUC	**0.001,0.001**	0.68945	0.72379	0.70210	0.60491	0.71509	**0.68707**
	0.001,0.0001	0.69955	0.63207	0.66824	0.64760	0.70600	0.67069
	0.0001,0.001	0.71260	0.67030	0.69543	0.60410	0.69182	0.67485
	0.0001,0.0001	0.65355	0.58651	0.68318	0.61898	0.68236	0.64492
AUPRC	**0.001,0.001**	0.71989	0.75527	0.71416	0.60987	0.73035	**0.70591**
	0.001,0.0001	0.70599	0.64443	0.65682	0.63108	0.70800	0.66926
	0.0001,0.001	0.72601	0.67776	0.69473	0.62028	0.71346	0.68645
	0.0001,0.0001	0.67018	0.62215	0.67806	0.59238	0.68787	0.65013
F1	**0.001,0.001**	0.47924	0.55155	0.61436	0.17278	0.38334	**0.44025**
	0.001,0.0001	0.29184	0.23960	0.35431	0.22623	0.27565	0.27753
	0.0001,0.001	0.57188	0.43488	0.60112	0.23014	0.27277	0.42216
	0.0001,0.0001	0.37040	0.30783	0.49326	0.17507	0.23571	0.31646
ACC	**0.001,0.001**	0.59250	0.63077	0.64217	0.51174	0.61619	**0.59867**
	0.001,0.0001	0.56300	0.51462	0.56609	0.52870	0.56667	0.54781
	0.0001,0.001	0.63650	0.59462	0.63739	0.52696	0.57714	0.59452
	0.0001,0.0001	0.57250	0.53308	0.60478	0.51174	0.55810	0.55604
MCC	**0.001,0.001**	0.22814	0.29291	0.29498	0.03906	0.24172	**0.21936**
	0.001,0.0001	0.13358	0.03524	0.14494	0.07690	0.13591	0.10531
	0.0001,0.001	0.30108	0.22377	0.28267	0.08754	0.16181	0.21138
	0.0001,0.0001	0.18258	0.08130	0.22832	0.01512	0.10948	0.12336
Recall	**0.001,0.001**	0.46400	0.50462	0.59565	0.17913	0.32600	**0.41388**
	0.001,0.0001	0.26300	0.30000	0.33652	0.24261	0.28400	0.28523
	0.0001,0.001	0.55300	0.36923	0.58696	0.20870	0.25800	0.39518
	0.0001,0.0001	0.32100	0.26615	0.44000	0.14783	0.21200	0.27740
Precision	**0.001,0.001**	0.72714	0.74439	0.66958	0.74587	0.81112	0.73962
	0.001,0.0001	0.78529	0.79538	0.75173	0.74499	0.81499	**0.77848**
	0.0001,0.001	0.71546	0.72707	0.66364	0.74835	0.81981	0.73486
	0.0001,0.0001	0.75445	0.65060	0.67531	0.63791	0.77086	0.69783

the average Precision by 3.958 %, and the average Recall by 3.017 %. Replacing the VAE with a standard autoencoder while removing the masking strategy (Code-adv) led to further reductions in the average AUC, AUPRC, MCC, ACC, and Precision compared to MVAEDA w/o mask, with decreases of 6.431 %, 3.7 %, 21.292 %, 9.095 %, and 9.294 %, respectively. Not using adversarial training (MVAEDA w/o adv), i.e., not training with WGAN-GP, caused the average AUC, AUPRC, MCC, ACC, and Recall to decrease by 1.585 %, 6.407 %, 11.443 %, 5.332 %, and 19.212 %, respectively. Compared to the full model, not using orthogonal loss (MVAEDA w/o ortho) resulted in declines across all average metrics, with the average AUC, AUPRC, F1, MCC, ACC, Recall, and Precision decreasing by 0.954 %, 1.43 %, 1.531 %, 0.542 %, 0.298 %, 4.952 %, and 0.803 %, respectively. These results highlight the effectiveness of the masking strategy, VAE, adversarial training, and orthogonal loss in MVAEDA.

4.2.4. Parameter analysis

We conducted a parameter analysis on the model's masking rate,

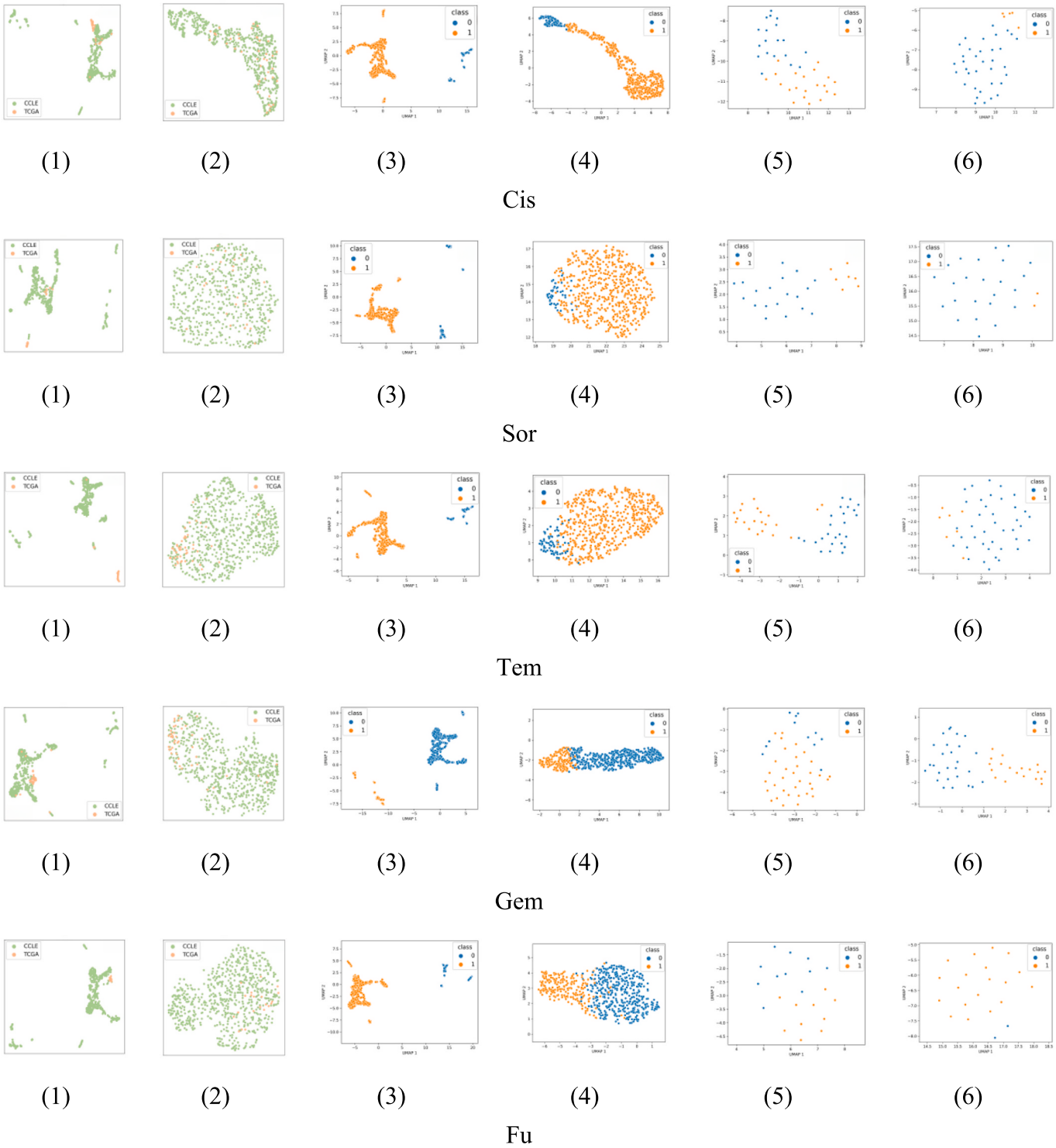


Fig. 3. UMAP visualization results. (1) represents the distribution of the original data; (2) represents the distribution of domain-invariant features learned by MVAEDA. (3–6) represent the initial features and learned embedding features of the drug's cell line data, the initial features and learned embedding features of the drug's patient data.

parameters α and λ in the loss function calculation, as well as the learning rate, to investigate the impact of different parameters on the model's performance. These analyses were all performed on the TCGA dataset. Table 6 shows that a low masking rate may fail to effectively learn complex features and contextual relationships, while a high masking rate may result in insufficient information in the input, making it difficult for the model to capture contextual information or reconstruct the complete content, thereby reducing the model's performance. Based on this analysis, we set the masking rate to 40 %. Table 7 indicates that overly emphasizing either predicting masked positions or reconstruction may decrease the model's performance. When $\alpha = 0.7$, the highest values across all average metrics are achieved. Therefore, we set $\alpha = 0.7$. Table 8 indicates that lower or higher weights lead to a decline in model performance. Lower weights may result in insufficient learning signals for masked samples, preventing the model from fully utilizing the information of the masked samples. Higher weights, on the other hand,

Table 10
Clustering results of drug cell line data and patient data: original and learned features.

Drug		Cell			Patient		
		Sihouette	DBI	NMI	Sihouette	DBI	NMI
Cis	Initial features	0.20081	1.78684	**0.07254**	0.14360	2.15607	0.15205
	Learned embeddings	**0.77217**	**0.58945**	0.05119	**0.75154**	**0.56652**	**0.17871**
Sor	Initial features	0.19925	**1.79596**	**0.12494**	0.28761	1.38829	**0.15961**
	Learned embeddings	**0.62739**	1.96519	0.07518	**0.53468**	**0.82861**	0.11724
Tem	Initial features	0.20008	**1.79592**	**0.13782**	0.24457	**1.59293**	**0.06863**
	Learned embeddings	**0.56075**	1.82806	0.03579	**0.27834**	1.77332	0.03028
Gem	Initial features	0.20083	1.78696	**0.06374**	**0.29373**	1.87504	0.00777
	Learned embeddings	**0.45870**	**1.14313**	0.00782	0.24769	**1.29075**	**0.01283**
Fu	Initial features	**0.20101**	**1.78787**	**0.09966**	0.09211	2.36416	0.01535
	Learned embeddings	0.17191	2.09089	0.04991	**0.18268**	**1.33097**	**0.13141**

may cause the model to overly focus on the masked samples, leading to a deviation from the overall data distribution. Therefore, we set $\lambda = 0.75$. In Table 9, the "Lr" column contains two numbers: the first represents the learning rate during non-adversarial training, and the second represents the learning rate during adversarial training. The results show that when the learning rates are set to (0.001, 0.001), the average AUC, AUPRC, F1, MCC, ACC, and Precision achieve their highest values, while Recall achieves the second-highest value. Therefore, we set the learning rates to (0.001, 0.001).

4.2.5. UMAP visualization

We employed UMAP tool to visualize and cluster drugs and cell-lines based on embeddings generated by MVAEDA. For each drug in TCGA dataset, we map both cell lines and patients based on their initial and domain-invariant feature learned by MVAEDA. For each drug, Fig. 3 (1) illustrates the distribution of the original data, with green dots representing CCLE samples and orange dots representing TCGA samples. The clear separation between these points highlights the distribution differences between the two domains in their original feature space. Fig. 3 (2) displays the domain-invariant features of cell lines and patient samples generated by our model, demonstrating significant overlap between the two domains. This indicates that the generated embeddings effectively reduced the distribution differences between the two domains. Furthermore, we used k-means clustering to group the cell lines or patients based on their original or domain-invariant features. Fig. 3 (3–6) shows how cell lines or patients can be divided into two groups, representing sensitivity and resistance to drugs, based on their original or domain-invariant features. We also calculated clustering evaluation metrics, including the Silhouette Coefficient (SC), Davies-Bouldin Index (DBI), and Normalized Mutual Information (NMI). Table 10 demonstrates that the embedded features effectively grouped the cell line data and patient data into two clusters, achieving both compactness within clusters and separability between them. This suggests the domain-invariant features' ability to distinguish between sensitive and resistant group to a certain extent. Overall, the results indicate that the model possesses discrimination capability. However, the performance may be limited by the lack of joint training between the classifier and feature extractor.

5. Conclusion

This work introduces MVAEDA, a novel method for predicting anticancer drug response in patients based on masked variational autoencoders. Unlike traditional transfer learning methods, MVAEDA employs variational autoencoders (VAEs) and a mask strategy with a mask predictor to learn domain-invariant latent representations. We evaluate it on a selected dataset from the clinical TCGA dataset and the preclinical PDTC dataset. For most drugs, its AUC, AUPRC, MCC, ACC and Precision values are higher than other existing methods. In addition, we input the features extracted by our method into DeepDCR and DeepDSC, which can effectively improve the performance of the model, indicating the effectiveness of our feature extraction. Finally, we show through ablation experiments that our mask strategy and the use of variational autoencoder (VAE) help improve model performance. However, our model still needs to be improved. The features of drugs can effectively distinguish different drugs. The fusion of drug features may better represent the patient's response to different drugs, and a good downstream task design can effectively improve the performance of the model. Therefore, in the future, we will consider improving the model by reasonably adding drug features and designing good downstream tasks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China under grant No.62472202, No.61972185 and Yunnan Ten Thousand Talents Plan young.

Availability and implementation.

The data and source code are available at: https://github.com/weiba/MVAEDA

Data availability

Data will be made available on request.

References

[1] J. Barretina, G. Caponigro, N. Stransky, K. Venkatesan, A.A. Margolin, S. Kim, L. A. Garraway, The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity, Nature 483 (7391) (2012) 603–607, https://doi.org/10.1038/nature11003.

[2] W. Yang, J. Soares, P. Greninger, E.J. Edelman, H. Lightfoot, S. Forbes, M. J. Garnett, Genomics of Drug Sensitivity in Cancer (GDSC): a resource for therapeutic biomarker discovery in cancer cells, Nucleic Acids Res. 41 (D1) (2012) D955–D961, https://doi.org/10.1093/nar/gks1111.

[3] M. Li, Y. Wang, R. Zheng, X. Shi, Y. Li, F.X. Wu, J. Wang, DeepDSC: A Deep Learning Method to Predict Drug Sensitivity of Cancer Cell Lines, IEEE/ACM Trans. Comput. Biol. Bioinf. 18 (2) (2021) 575–582, https://doi.org/10.1109/TCBB.2019.2919581.

[4] Q. Liu, Z. Hu, R. Jiang, M. Zhou, DeepCDR: a hybrid graph convolutional network for predicting cancer drug response, Bioinformatics 36 (Supplement_2) (2020) i911–i918, https://doi.org/10.1093/bioinformatics/btaa822.

[5] P. Li, Z. Jiang, T. Liu, X. Liu, H. Qiao, X. Yao, Improving drug response prediction via integrating gene relationships with deep learning, Brief. Bioinform. 25 (3) (2024) bbae153, https://doi.org/10.1093/bib/bbae153.

[6] Q. Wei, S.A. Ramsey, Predicting chemotherapy response using a variational autoencoder approach, BMC Bioinf. 22 (2021) 1–22, https://doi.org/10.1186/s12859-021-04339-6.

[7] W. Peng, T. Chen, W. Dai, Predicting drug response based on multi-omics fusion and graph convolution, IEEE J. Biomed. Health Inform. 26 (3) (2021) 1384–1393, https://doi.org/10.1109/JBHI.2021.3102186.

[8] W. Peng, H. Liu, W. Dai, N. Yu, J. Wang, Predicting cancer drug response using parallel heterogeneous graph convolutional networks with neighborhood

interactions, Bioinformatics 38 (19) (2022) 4546–4553, https://doi.org/10.1093/bioinformatics/btac574.
[9] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, Q. He, A comprehensive survey on transfer learning, Proc. IEEE 109 (1) (2020) 43–76, https://doi.org/10.1109/JPROC.2020.3004555.
[10] J. Ma, S.H. Fong, Y. Luo, C.J. Bakkenist, J.P. Shen, S. Mourragui, T. Ideker, Few-shot learning creates predictive models of drug response that translate from high-throughput screens to individual patients, Nature Cancer 2 (2) (2021) 233–244, https://doi.org/10.1038/s43018-020-00169-2.
[11] J. Chen, X. Wang, A. Ma, Q.E. Wang, B. Liu, L. Li, Q. Ma, Deep transfer learning of cancer drug responses by integrating bulk and single-cell RNA-seq data, Nat. Commun. 13 (1) (2022) 6494, https://doi.org/10.1038/s41467-022-34277-7.
[12] D. He, Q. Liu, Y. Wu, L. Xie, A context-aware deconfounding autoencoder for robust prediction of personalized clinical drug response from cell-line compound screening, Nat. Mach. Intell. 4 (10) (2022) 879–892, https://doi.org/10.1038/s42256-022-00541-0.
[13] H. Sharifi-Noghabi, P.A. Harjandi, O. Zolotareva, C.C. Collins, M. Ester, Out-of-distribution generalization from labelled and unlabelled gene expression data for drug response prediction, Nat. Mach. Intell. 3 (11) (2021) 962–972, https://doi.org/10.1038/s42256-021-00408-w.
[14] R. Peres da Silva, C. Suphavilai, N. Nagarajan, TUGDA: task uncertainty guided domain adaptation for robust generalization of cancer drug response prediction from in vitro to in vivo settings, Bioinformatics 37 (Supplement_1) (2021) i76–i83, https://doi.org/10.1093/bioinformatics/btab299.
[15] A.B. Dincer, J.D. Janizek, S.I. Lee, Adversarial deconfounding autoencoder for learning robust gene expression embeddings, Bioinformatics 36 (Supplement_2) (2020) i573–i582, https://doi.org/10.1093/bioinformatics/btaa796.
[16] D.P. Kingma, M. Welling (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114. https://doi.org/10.48550/arXiv.1312.6114.
[17] J. Devlin, M.W. Chang, K. Lee, K. Toutanova, (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://doi.org/10.48550/arXiv.1810.04805.
[18] Z. Fang, R. Zheng, M. Li, scMAE: a masked autoencoder for single-cell RNA-seq clustering, Bioinformatics 40 (1) (2024) btae020, https://doi.org/10.1093/bioinformatics/btae020.
[19] Z. Ding, S. Zu, J. Gu, Evaluating the molecule-based prediction of clinical drug responses in cancer, Bioinformatics 32 (19) (2016) 2891–2895, https://doi.org/10.1093/bioinformatics/btw344.
[20] E.A. Clayton, T.A. Pujol, J.F. McDonald, P. Qiu, Leveraging TCGA gene expression data to build predictive models for cancer drug response, BMC Bioinf. 21 (2020) 1–11, https://doi.org/10.1186/s12859-020-03690-4.
[21] A. Bruna, O.M. Rueda, W. Greenwood, A.S. Batra, M. Callari, R.N. Batra, C. Caldas, A biobank of breast cancer explants with preserved intra-tumor heterogeneity to screen anticancer compounds, Cell 167 (1) (2016) 260–274, https://doi.org/10.1016/j.cell.2016.08.041.
[22] M. Arjovsky, S. Chintala, L. Bottou, (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.
[23] P. Vincent, H. Larochelle, Y. Bengio, P.A. Manzagol, (2008, July). Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning (pp. 1096-1103). https://doi.org/10.1145/1390156.1390294.
[24] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, D. Erhan, (2016). Domain separation networks. Advances in neural information processing systems, 29.
[25] Z. Zheng, J. Chen, X. Chen, L. Huang, W. Xie, Q. Lin, K.C. Wong, Enabling Single-Cell Drug Response Annotations from Bulk RNA-Seq Using SCAD, Adv. Sci. 10 (11) (2023) 2204113, https://doi.org/10.1002/advs.202204113.