Transfer Learning for Predicting Gene Regulatory Effects of Chemicals

Bahattin Can Maral, Mehmet Tan

Transfer Learning for Predicting Gene Regulatory Effects of Chemicals

Bahattin Can Maral[0000−0002−1552−1592] and Mehmet Tan[0000−0002−1741−0570]

TOBB University of Economics and Technology, Sogutozu, Ankara, Turkey
{bahattincanmaral,mtan}@etu.edu.tr

Abstract. Among the recent developments in bioinformatics and chemogenomics, various deep learning methods have been the most prevalent [4, 5, 9]. This resulted in an over-saturation of powerful models that easily pushed the limits of existing datasets. Subsequently, many novel advancements have been done with improvements to the datasets. Amidst these advancements, researchers of Deep Compound Profiler (DeepCOP) [10] set themselves apart with a novel method of introducing new features whilst keeping the deep learning model relatively basic. In this study, we propose to take this novel method one step further by applying transfer learning between cell lines. In order to better evaluate the benefits of transfer learning, we've introduced 2 drug-based data splits. The transfer learning method, as its core, utilizes the learned knowledge of "source" cell lines to give a head start to "target" cell lines. Taking advantage of prior knowledge from source cell lines not only boosts existing compounds' effect prediction, but it can also be used as a premonition for the compounds' (explicit to the source), effects on target cell lines before they are tested in real life. Our experiments showed improvements up to 22.81% improvement on area under ROC curve (AUC) on the split closest to a wet lab experiment.

Keywords: Transfer learning · Chemogenomics · Domain adaptation.

1 Introduction

Chemogenomics [1] studies chemical compounds and their genomic/proteomic effects on biological systems.

Since machine learning, chemogenomics has been advancing rapidly, the ability to simulate chemical effects on different cell lines with almost no cost also helped the drug-discovery process by identifying potential side effects of the candidate drugs early.

DeepCOP is a recently proposed deep learning model that is trained on the LINCS L1000 [8] transcriptomics dataset to predict differential gene expression effects of chemicals. In the study, researchers train two deep learning models in the form of a multilayer perceptron (MLP) for each selected cell line.

The study distinguishes itself with the heavy focus on feature generation and uncomplicated model structure.

In this study, we aim to take the work done by DeepCOP researchers one step further by introducing transfer learning to utilize the knowledge of different cell lines.

In machine learning, transfer learning is defined as the use of previously acquired knowledge gained while solving one problem on a different but related problem. For this study, different problems correspond to different cell lines. Utilizing the data from other cell lines also comes with the advantage of training a more general model due to there being experiments with chemical compounds unique to some cell lines. In addition to transfer learning, we have introduced 2 splitting methods that we think better represents the high-throughput screening process and 17 previously discarded cell lines.

This paper includes 3 key sections: Materials and methods, discussion, and conclusion. In the first of which, we briefly explain preparation of the dataset, clarify the key differences between the splitting methods, and give average improvements for each method. The discussion section mainly focuses on splitting complications, what can be done to counter them, and what we did in this study. In the conclusion section, we wrap up the paper, and give a short summary for the planned studies moving forward.

2 Materials and methods

We aim to prove the usefulness of transfer learning for the prediction of chemical induced differential gene expression. To produce comparable results to DeepCOP, the data production and preparation steps have been replicated, and the novel ideas repeated with the original methodology.

2.1 Datasets

L1000 dataset provides the data in the form of levels. In this study, we utilize the Level 5 dataset. Level 5 data comprise the gene profiles of 978 landmark genes' drug-gene interaction experiments on distinct cell lines. To get to the Level 5 of the L1000 dataset, the same method of genex measurement, namely flow cytometry, is used throughout levels 1 to 5. At Level 5 these values are reformed into standardized z-scores. Further information about this dataset can be found in [8].

For data preparation, we've closely followed DeepCOP; with the only difference being the removal of the 10% threshold, for the sake of space limitations.

In DeepCOP, from the processed dataset, researchers selected the top six cell lines that had the largest number of drug-gene interaction experiments i.e., VCAP, A549, A375, PC3, MCF7, and HT29. For this study, we expanded this selection to every cell line that had more than 10 samples, which resulted in 17 additional cell lines compared to [10]. These additional cell lines are as follows: BT20, HA1E, HCC515, HEK293T, HEPG2, HL60, HS578T, HUH7, JURKAT, MCF10A, MDAMB231, NKDBA, NOMO1, SKBR3, THP1, U266, and U937 where details are given in Table 1.

Table 1. The details of the cell line id's that are included in this study.

Cell Line	Primary Site	Sample Type	Subtype
A375	skin	tumor	malignant melanoma
A549	lung	tumor	non small cell lung cancer carcinoma
BT20	breast	tumor	carcinoma
HA1E	kidney	normal	normal kidney
HCC515	lung	normal	carcinoma
HEK293T	kidney	normal	embryonal kidney
HEPG2	liver	tumor	hepatocellular carcinoma
HL60	haematopoietic, lymphoid tissue	tumor	acute myelogenous leukemia, promyelocytic
HS578T	breast	tumor	carcinoma
HT29	large intestine	tumor	colorectal adenocarcinoma
HUH7	liver	tumor	hepatocellular carcinoma
JURKAT	haematopoietic, lymphoid tissue	tumor	acute lymphoblastic leukemia, T-cell
MCF10A	breast	normal	epithelial
MCF7	breast	tumor	adenocarcinoma
MDAMB231	breast	tumor	adenocarcinoma
NKDBA	kidney	normal	kidney epithelial
NOMO1	haematopoietic, lymphoid tissue	tumor	acute myeloid leukemia
PC3	prostate	tumor	adenocarcinoma
SKBR3	breast	tumor	adenocarcinoma
THP1	haematopoietic, lymphoid tissue	tumor	acute myelogenous leukemia, monocytic
U266	blood	tumor	myeloman, haematopoietic, lymphoid
U937	haematopoietic, lymphoid tissue	tumor	lymphoma, B-cell, non-hodgkin's, histiocytic
VCAP	prostate	tumor	carcinoma

2.2 Feature generation

Similar to the well-known Extended-connectivity fingerprints (ECFPs) or Functional-class fingerprints (FCFPs), Morgan fingerprints are used to represent chemical structures as binary arrays. From the first time, it was introduced[3], to the modernized implementation in [7], Morgan fingerprints have been one of the most prevalent fingerprinting methods to this day. In this study, Morgan descriptors were generated using the RDKIT Open-source cheminformatics toolkit[6] to correlate the chemical structure to gene perturbation. The descriptors on the conventional SMILES form were computed with a radius of 2 to generate a one-hot vector of 2048 features for 19 811 compounds.

While there are no widely acknowledged techniques for describing a gene, the gene ontology (GO) consortium [2], with 40 thousand GO terms and over 200

000 qualitative annotations for Homo sapiens, is the most prominent. DeepCOP proposes a novel way to utilize the GO descriptors along with the 978 landmark genes. Using the OntologyX R package, GO terms that correlated with at least 3 landmark genes were extracted. Then each gene was described using the appropriate GO terms. One-hot encoding of these GO descriptors resulted in a binary array of size 1107 that can be used as features alongside the Morgan fingerprints.

2.3 Data splitting

In DeepCOP, researchers have split the aforementioned dataset into 10 randomly selected folds to perform 10-fold cross-validation for each cell line. This approach ensures every compound-gene combination is unique per fold. However, it does not accommodate the fact that each drug is repeated by the number of landmark genes. Conversely, each gene is repeated by the number of compounds. Therefore, a model trained on randomly split 9 folds, has already encountered the input elements (compound and gene) for the remaining fold, separately. This leads to complications that may cause over-fitting, for the reasons that we further addressed in the discussion section. For the sake of the completeness of this study, we have also repeated our experiments in this format which will be called "random-split" moving forward.

Cold-Drug splitting method divides the compound data into 10 folds, then populates each fold with the gene data. This ensures the drugs in the test fold are brand new for the trained models. We named this method of splitting "cold-drug-split". For transfer learning, the source model is trained on a 95%-5% random-split. Then we train the target cell line data on the cold-drug-split. We consider this splitting method to be the closest to real-life high-throughput testing.

Similar to cold-drug-split, we introduced Transfer-Drug splitting to counteract contamination of the test data. The main difference between the two splits is that this splitting method also eliminates the contamination between cell lines while transfer learning. On the last step of transfer learning on cold-drug-split, the compounds used in the training of the source model are removed from the test fold. This guarantees that the compounds in the test data are never-before-seen for the model. Going forward, this splitting method will be referred to as "transfer-drug-split".

2.4 Experiments

This study aims to justify the value of utilizing previously tested knowledge of cell lines in novel test domains. To be able to make a direct comparison with DeepCOP results, we have repeated the original experiments both with a random split and a transfer-drug split as a baseline. During these experiments, we also discovered that similar scores could be achieved with much smaller neural networks from the original study. The results of the experiments listed below have been achieved with a neural network that reduces the neuron count of the original 2 hidden layers to 400 (from 3155 of DeepCOP), while the rest of the network structure remains unchanged.

Based on the assumption that learned knowledge from cell lines could benefit others, we've opted to use network-based parameter sharing as our transfer learning method. Parameter sharing methods, transfer their learned knowledge between problems by keeping a part of or the whole network trained on the donor/source problem. Continuing the training on the receiver/target problem on the transferred network by either modifying the received weights or adding brand-new layers to train.

In this study, we trained a source model for every single cell line with 95% train and 5% validation split data. Then we saved the finalized weights of the source model and used them as the initial weights for training the target cell line model. We repeated this process for every cell line pair. Network structure and parameters were kept unchanged throughout experiments to ensure comparability. We selected the original six cell lines of the DeepCOP study as our source models because there was a huge falloff in the sample count after these six. The more knowledge we can acquire in the source problem makes for a stronger baseline for the target.

As our first objective, we trained target cell lines on the same random-split with DeepCOP. Evaluating our model on random-split resulted in average improvements of 4.51% for up-regulated and 4.52% for down-regulated models' AUC scores. Among up-regulated models, the maximum recorded betterment was on the NKDBA cell line when MCF7 was used as a source; which resulted in 15.73% improvement of the AUC score. As for down-regulated models, the maximum improvement was on the HL60 cell line with A375 as the source; that resulted in an added 12.73% for the AUC score.

A detailed view of the random-split experiments can be seen on the Table 2, for up and down-regulated model results. Each row represents the different target cell lines, and each column corresponds to different sources, while the "No-TL" (no transfer learning) column represents the results of the models when the model is trained on the same split, without a source model, on target cell line data. When the data is viewed as a whole, a clear trend emerges: The least amount of improvements are seen on the experiments that are trying to improve upon the cell lines we've selected as the source. This directly correlates to the number of samples each cell line possesses. For the remaining 17 target cell lines, the average gains for the AUC scores are 6.09% and 6.04% for up and down regulation models. While the scores are impressive, the random-split is the least realistic of the three. Transcriptomics datasets, like L1000, are subsets of gene perturbation measurements of wet-lab experiments. In these experiments, a chemical compound is tested on a cell line and the perturbations of the genes are measured simultaneously. Therefore, predicting a gene's perturbation by utilizing other gene's responses, is only useful in a situation where that specific gene perturbation value is lost. Which isn't useful for simulating high-throughput screening.

Apart from being unrealistic, this became a visible issue while we were trying to improve DeepCOP's scores on isolation with network optimization, for a stronger baseline. Our experiments showed minimal change in evaluation, even

Table 2. AUC scores of regulated genes on random-drug-split 10-fold data, rows and columns indicate target and source cell lines. The best results in a row are indicated in bold.

		Source Cell						
	Target Cell ↓	**No-TL**	**A375**	**A549**	**HT29**	**MCF7**	**PC3**	**VCAP**
Up	**A375**	**0.8199**		0.8189	0.8190	0.8141	0.8167	0.8105
	A549	**0.8254**	0.8243		0.8228	0.8246	0.8244	0.8222
	BT20	0.7698	0.7975	**0.8015**	0.7994	0.8011	0.7987	0.7944
	HA1E	0.8281	**0.8298**	0.8293	0.8281	0.8260	0.8297	0.8251
	HCC515	0.8397	0.8405	0.8400	0.8403	0.8384	**0.8413**	0.8392
	HEK293T	0.8011	0.8549	0.8533	0.8431	0.8503	**0.8576**	0.8495
	HEPG2	0.8216	0.8548	0.8572	**0.8584**	0.8534	0.8517	0.8412
	HL60	0.8427	**0.8964**	0.8836	0.8928	0.8872	0.8852	0.8835
	HS578T	0.7556	0.7901	**0.7976**	0.7901	0.7909	0.7932	0.7859
	HT29	0.7904	**0.7966**	0.7902		0.7875	0.7899	0.7822
	HUH7	0.7505	0.8038	0.7998	**0.8072**	0.8048	0.8032	0.7987
	JURKAT	0.8071	0.8579	0.8493	**0.8602**	0.8029	0.8536	0.8452
	MCF10A	0.7733	**0.8054**	0.8002	0.8017	0.8013	0.7988	0.7928
	MCF7	0.8313	0.8291	0.8315	0.8293		**0.8327**	0.8295
	MDAMB231	0.7662	0.8207	0.8234	0.8214	0.8212	**0.8258**	0.8176
	NKDBA	0.6833	0.7876	0.7891	0.7864	**0.7908**	0.7891	0.7878
	NOMO1	0.7828	0.8013	**0.8278**	0.7485	0.7840	0.7816	0.7731
	PC3	**0.8327**	0.8253	0.8276	0.8248	0.8279		0.8268
	SKBR3	0.7655	0.7868	0.7943	0.7918	**0.7991**	0.7933	0.7855
	THP1	0.7296	0.7492	0.7986	0.7805	0.7887	0.7883	**0.8130**
	U266	0.7898	0.8146	0.8288	**0.8327**	0.7948	0.8307	0.8308
	U937	0.7572	0.8023	0.8019	**0.8085**	0.7721	0.7993	0.7898
	VCAP	**0.8471**	0.8457	0.8458	0.8450	0.8469	0.8466	
Down	**A375**	0.8239		0.8244	**0.8248**	0.8214	0.8201	0.8165
	A549	**0.8154**	0.8152		0.8132	0.8137	0.8153	0.8103
	BT20	0.7758	0.8187	0.8173	0.8164	**0.8267**	0.8225	0.8102
	HA1E	0.8384	**0.8403**	0.8380	0.8397	0.8382	0.8400	0.8342
	HCC515	0.8420	0.8444	0.8440	0.8449	0.8448	**0.8473**	0.8423
	HEK293T	0.8383	0.8835	0.8797	0.8766	0.8838	**0.8864**	0.8739
	HEPG2	0.8341	0.8635	0.8638	0.8636	**0.8645**	0.8600	0.8507
	HL60	0.7917	**0.8925**	0.8744	0.8881	0.8737	0.8739	0.8803
	HS578T	0.7843	0.8080	**0.8082**	0.8037	0.8063	0.8077	0.7993
	HT29	0.7974	**0.8032**	0.7954		0.7932	0.7960	0.7915
	HUH7	0.7682	0.7954	0.7953	**0.7988**	0.7963	0.7957	0.7816
	JURKAT	0.7917	**0.8781**	0.8506	0.8701	0.8591	0.8714	0.8422
	MCF10A	0.7955	0.8200	0.8192	0.8196	**0.8215**	0.8165	0.8178
	MCF7	**0.8430**	0.8425	0.8415	0.8414		0.8430	0.8395
	MDAMB231	0.7953	0.8395	0.8367	0.8345	**0.8431**	0.8397	0.8335
	NKDBA	0.7241	0.7948	0.7983	0.7914	**0.8002**	0.7967	0.7926
	NOMO1	0.7934	0.8044	0.7665	0.7420	**0.8467**	0.8044	0.6887
	PC3	0.8327	0.8322	0.8323	0.8314	**0.8345**		0.8320
	SKBR3	0.7951	0.8233	0.8225	0.8255	**0.8292**	0.8203	0.8208
	THP1	0.7590	0.7945	0.8156	0.7439	**0.8399**	0.7845	0.7082
	U266	0.7765	0.8192	0.8076	0.8217	0.8344	0.8241	**0.8345**
	U937	0.7569	**0.8050**	0.7642	0.8025	0.7370	0.7168	0.7705
	VCAP	0.8564	0.8554	0.8541	0.8539	**0.8570**	0.8561	

for major changes in the network structure. Upon further investigation, we deduced that the almost static but low error rate was limited by the data itself. While producing impressive results, the neural network itself wasn't modeling the data correctly, and instead was basically training on the test data, due to the repetitions in the dataset. This was also the reason behind the change in the layer sizes since smaller layers could produce similar results faster.

To eliminate the compound repetition, we've introduced the cold-drug-split. Repeating the original methodology with this new split resulted in 23.05% lower AUC scores on average. These results can be seen on the No-TL columns of the Table 3. The most improvement for the up-regulated models was on the JURKAT cell line when A375 cell line data was used as the source; which boosted the AUC score by 22.81%. For down-regulated, it was on the HCC515 cell line by 14.38% when the PC3 was used as the source. The results of transfer learning experiments on cold-drug-split can be seen on Table 3. As mentioned before, we regard cold-drug-split as the most realistic method of the three. Experiments on this split resulted in average improvements of 9.70% and 8.29% on AUC scores for up and down-regulated models, respectively.

The second splitting method we introduced, transfer-drug-split, while a bit unrealistic compared to cold-drug-split, posed an important challenge to the model training. We've re-tested the models we trained in cold-drug-split by removing the compounds that also were tested on the source cell line from the target test splits, producing a testing environment in which the model was tested purely tested on its ability to process new compounds. On transfer-drug-split, the AUC scores on average improved by 1.42% and -0.73% for up and down-regulated models. However, the maximum AUC gains were comparable to the ones on the other splits. For up-regulated, training the JURKAT cell line on top of the A375 resulted in 19.88% improved AUC. Among down-regulated it was HS578T which improved its AUC score by 6.74% when trained after MCF7.

Detailed AUC scores of transfer-drug-split experiments can be seen on Table 4, for up and down-regulated model results.

3 Discussion

3.1 Best Improvements

When we look at the top 10 most improved AUC scores in Table 5, several trends emerge that we can comment on.

Firstly, every experiment on the table is an up differential expression (DE) model. If we were to take an average of every up, and down differential expression experiments separately, we can see the up regulated experiments are more effected than down regulated ones. This can be correlated to the greater up regulated sample counts compared to the downs.

Another visible trend is in the splits. Other than one exception, every split in the table is either from a cold-drug-split or a transfer-drug-split experiment. This is mainly because of the 23% decrease in the AUC values when we'd switched

Table 3. AUC scores of regulated genes on cold-drug-split 10-fold data, rows and columns indicate target and source cell lines. The best results in a row are indicated in bold.

		Source Cell						
	Target Cell ↓	**No-TL**	**A375**	**A549**	**HT29**	**MCF7**	**PC3**	**VCAP**
Up	**A375**	0.6145		0.6633	0.6499	0.6558	**0.6652**	0.6304
	A549	0.5864	0.5944		0.5851	0.6294	**0.6445**	0.6145
	BT20	0.5952	0.5460	0.6316	0.5801	**0.6553**	0.6268	0.6013
	HA1E	0.6220	0.6601	0.6701	0.6482	0.6627	**0.6859**	0.6666
	HCC515	0.6136	0.6401	0.6761	0.6507	0.6679	**0.6970**	0.6664
	HEK293T	0.6165	**0.6971**	0.6238	0.6698	0.6543	0.6348	0.6466
	HEPG2	0.6096	0.7297	0.7192	0.6761	0.6893	**0.7388**	0.6224
	HL60	0.6063	0.6183	0.5397	**0.6665**	0.5670	0.5671	0.5604
	HS578T	0.5762	0.5796	0.6509	0.6306	**0.6725**	0.6436	0.6307
	HT29	0.6197	**0.6892**	0.6756		0.6742	0.6719	0.6374
	HUH7	0.6428	0.6197	0.6856	0.6488	0.6749	**0.6980**	0.6679
	JURKAT	0.5558	**0.6826**	0.5527	0.5816	0.5898	0.6307	0.6205
	MCF10A	0.6434	0.6131	0.6360	0.6291	**0.6594**	0.6460	0.6323
	MCF7	0.6059	0.6151	0.6569	0.6100		**0.6752**	0.6477
	MDAMB231	0.6191	0.5859	0.6300	0.6138	**0.6896**	0.6586	0.6288
	NKDBA	0.6121	0.5950	0.6267	0.6130	**0.6647**	0.6091	0.6129
	NOMO1	0.5888	0.5100	0.5758	0.5689	0.5590	**0.5920**	0.4933
	PC3	0.6055	0.6153	0.6453	0.6074	**0.6644**		0.6386
	SKBR3	0.6057	0.5599	0.6161	0.5757	**0.6575**	0.6311	0.5835
	THP1	0.5849	0.5427	0.5922	0.5841	**0.6019**	0.5848	0.5697
	U266	**0.6430**	0.6557	0.6408	0.6136	0.6035	0.6249	0.5553
	U937	0.5908	0.5800	0.6002	**0.6067**	0.5710	0.6046	0.5772
	VCAP	0.5956	0.5970	0.6238	0.5924	0.6281	**0.6387**	

Down	**A375**	0.6228		0.6648	0.6575	0.6563	**0.6671**	0.6317
	A549	0.5799	0.5884		0.5933	0.6309	**0.6352**	0.6162
	BT20	0.6236	0.6222	0.6620	0.6303	**0.7064**	0.6697	0.6188
	HA1E	0.6331	0.6685	0.6673	0.6505	0.6693	**0.6954**	0.6723
	HCC515	0.6184	0.6440	0.6836	0.6488	0.6885	**0.7073**	0.6776
	HEK293T	0.6982	0.7408	0.7752	0.7292	**0.7781**	0.6995	0.7610
	HEPG2	0.6653	**0.7457**	0.7450	0.7039	0.7415	0.7421	0.6554
	HL60	**0.7314**	0.6375	0.6507	0.6811	0.6235	0.6327	0.6659
	HS578T	0.6121	0.6203	0.6621	0.6469	**0.6964**	0.6575	0.6374
	HT29	0.6199	**0.6969**	0.6709		0.6742	0.6718	0.6396
	HUH7	0.6469	0.6543	0.6766	0.6777	**0.6955**	0.6708	0.6783
	JURKAT	0.6679	**0.7213**	0.6766	0.6820	0.5887	0.6649	0.6359
	MCF10A	0.6355	0.6396	0.6510	0.6559	**0.6805**	0.6800	0.6469
	MCF7	0.6148	0.6240	0.6600	0.6207		**0.6950**	0.6575
	MDAMB231	0.6595	0.6323	0.6702	0.6395	**0.7080**	0.6845	0.6370
	NKDBA	0.6320	0.6387	0.6364	0.6289	**0.6673**	0.6345	0.6252
	NOMO1	0.5725	0.5049	0.5858	0.5596	0.5649	**0.6268**	0.5099
	PC3	0.6118	0.6166	0.6526	0.6209	**0.6740**		0.6489
	SKBR3	0.6472	0.6325	0.6320	0.6474	**0.6937**	0.6545	0.6116
	THP1	0.6326	0.5687	0.5944	0.5802	**0.6413**	0.6338	0.5806
	U266	0.6799	0.6907	0.7053	0.6524	0.6506	**0.7065**	0.5438
	U937	0.5641	0.5860	0.5903	0.5786	0.5697	0.5750	**0.5913**
	VCAP	0.5965	0.6014	0.6197	0.6007	0.6272	**0.6458**	

Table 4. AUC scores of regulated genes on transfer-drug-split 10-fold data, rows and columns indicate target and source cell lines. The best results in a row are indicated in bold.

		Source Cell						
	Target Cell ↓	**No-TL**	**A375**	**A549**	**HT29**	**MCF7**	**PC3**	**VCAP**
Up	**A375**	**0.6145**		0.6065	0.5943	0.5933	0.6060	0.5767
	A549	**0.5864**	0.5414		0.5354	0.5711	0.5780	0.5572
	BT20	0.5952	0.5156	0.5765	0.5440	**0.6019**	0.5729	0.5409
	HA1E	0.6220	0.5979	0.6096	0.5913	0.6059	**0.6252**	0.6085
	HCC515	0.6136	0.5843	0.6164	0.5899	0.6155	**0.6381**	0.6169
	HEK293T	0.6165	**0.6302**	0.5973	0.5891	0.5685	0.5517	0.5673
	HEPG2	0.6096	0.6504	0.6458	0.6255	0.6302	**0.6517**	0.5811
	HL60	0.6063	0.5978	0.5479	**0.6268**	0.5572	0.5543	0.5590
	HS578T	0.5762	0.5426	0.5796	0.5746	**0.6263**	0.5851	0.5655
	HT29	0.6197	**0.6223**	0.6076		0.6028	0.5999	0.5781
	HUH7	**0.6428**	0.5984	0.6334	0.6157	0.6274	0.6221	0.5978
	JURKAT	0.5558	**0.6663**	0.6485	0.6325	0.6208	0.6652	0.6300
	MCF10A	**0.6434**	0.5687	0.5845	0.5827	0.6160	0.6005	0.5722
	MCF7	0.6059	0.5612	0.5896	0.5569		**0.6139**	0.5857
	MDAMB231	0.6191	0.5545	0.5905	0.5852	**0.6397**	0.6074	0.5700
	NKDBA	0.6121	0.5408	0.5623	0.5565	**0.6205**	0.5665	0.5797
	NOMO1	**0.5888**	0.5472	0.4782	0.4862	0.4978	0.5132	0.4282
	PC3	0.6055	0.5619	0.5874	0.5554	**0.6061**		0.5835
	SKBR3	0.6057	0.5306	0.5651	0.5460	**0.6196**	0.5798	0.5416
	THP1	**0.5849**	0.5095	0.5489	0.5360	0.5417	0.5522	0.4967
	U266	0.6430	**0.6619**	0.6252	0.5920	0.5915	0.6056	0.5308
	U937	**0.5968**	0.5570	0.5632	0.5792	0.5665	0.5739	0.5360
	VCAP	**0.5956**	0.5446	0.5680	0.5419	0.5755	0.5845	

Down	**A375**	**0.6228**		0.6085	0.6005	0.5962	0.6085	0.5765
	A549	**0.5799**	0.5376		0.5403	0.5727	0.5719	0.5594
	BT20	0.6236	0.5764	0.5992	0.5851	**0.6353**	0.6149	0.5696
	HA1E	0.6331	0.6029	0.6081	0.5890	0.6085	**0.6337**	0.6155
	HCC515	0.6184	0.5854	0.6166	0.5870	0.6239	**0.6455**	0.6270
	HEK293T	**0.6982**	0.6496	0.5940	0.5941	0.6015	0.6014	0.6125
	HEPG2	0.6653	**0.6693**	0.6693	0.6476	0.6639	0.6611	0.6072
	HL60	**0.7314**	0.6355	0.6215	0.6722	0.6274	0.6164	0.6266
	HS578T	0.6121	0.5725	0.6025	0.5900	**0.6534**	0.6029	0.5771
	HT29	0.6199	**0.6274**	0.6050		0.6046	0.6039	0.5761
	HUH7	**0.6469**	0.6113	0.6150	0.6275	0.6397	0.6126	0.6026
	JURKAT	0.7107	**0.7162**	0.6795	0.6728	0.6581	0.6953	0.6088
	MCF10A	**0.6355**	0.5980	0.6067	0.6163	0.6281	0.6200	0.6011
	MCF7	0.6148	0.5686	0.5962	0.5646		**0.6200**	0.5938
	MDAMB231	0.6595	0.5882	0.6082	0.6025	**0.6615**	0.6188	0.5855
	NKDBA	**0.6320**	0.5843	0.5739	0.5785	0.6219	0.5668	0.5589
	NOMO1	**0.5725**	0.4693	0.5309	0.4847	0.5083	0.5448	0.4489
	PC3	0.6118	0.5634	0.5925	0.5655	**0.6156**		0.5940
	SKBR3	0.6472	0.5858	0.5971	0.6041	**0.6495**	0.6131	0.5767
	THP1	**0.6326**	0.5500	0.5382	0.5443	0.5967	0.5946	0.5165
	U266	**0.6799**	0.6535	0.6438	0.5995	0.6150	0.6452	0.5164
	U937	0.5641	0.5605	0.5575	0.5747	**0.5837**	0.5476	0.5779
	VCAP	**0.5965**	0.5462	0.5637	0.5464	0.5753	0.5888	

to cold-drug-split. It is easier to improve upon on a worse baseline. Also, the complications in the random-drug-split makes it harder to improve its scores, of which we will discuss in the next section.

Lastly, two target cell lines dominate the top 10: JURKAT, and HEPG2. Interestingly, the reason behind them are polar opposites. For HEPG2, similar improvements are not visible in the transfer-drug-split experiments, therefore the benefits of transfer learning must be coming from the same drugs that are also tested on the source cell line. For the JURKAT cell line, the average improvement goes up by 63% when we switch to transfer-drug-split, which eliminates drugs tested on the source cell line. This points to better generalization for the target, and greater transferred knowledge from the source cell line.

Table 5. Transfer learning experiments that showed the most improvements for the direction of differential expression (DE) and split.

	Source - Target	**Improvement (DE, split)**
1	A375 - JURKAT	%22.81 (Up, cold)
2	PC3 - HEPG2	%21.19 (Up, cold)
3	A375 - JURKAT	%19.88 (Up, transfer)
4	A375 - HEPG2	%19.70 (Up, cold)
5	A549 - JURKAT	%19.69 (Up, transfer)
6	PC3 - JURKAT	%19.68 (Up, transfer)
7	A549 - HEPG2	%17.98 (Up, cold)
8	MCF7 - HS578T	%16.71 (Up, cold)
9	A549 - JURKAT	%16.69 (Up, transfer)
10	MCF7 - NKDBA	%15.73 (Up, random)

3.2 Complications of Using Random Split

The main idea behind DeepCOP is simple, yet effective: Utilizing differentiable characteristics of outputs lets us treat multiple genes' perturbation values as a single objective. This method populates the amount of data for each compound, retains the relation between individual genes, and gives each output a new meaning. However, for this theory to simulate real-life occurrences, the population should be done after the required validation split. To further demonstrate as an example; the THP1 cell line included in this study contains 18 different compounds. From the original data, we get 978 gene perturbations for each compound. After populating this data, we end up with 17,604 samples. However, we only have 18 unique compounds and 978 unique gene descriptors. Splitting this dataset into 10 random folds almost guarantees the possibility of data leakage.

There are two causes for this data leakage; duplicate genes, and compounds. In the L1000 dataset, an experiment corresponds to one compound, and its gene perturbation effects on a cell line. After the data processing step, we have ended up with the same compound repeated in 978 samples to predict each gene

separately. Randomly splitting this data means separating the gene outputs of a single experiment into different folds. It also implies that our model is learning to predict individual gene perturbations from the outcomes of other genes, for the same compound. This is an unrealistic and trivial task, since every gene perturbation is measured simultaneously for that experiment.

To counteract the data leakage caused by gene repetition, we would have to implement a transfer-gene split, where we would split 978 genes into different folds and populate the folds with the compounds. However, this splitting method is the polar opposite of the objective of HTS, since we're trying to simulate an untested compound's effects on existing genes and not the other way around. Therefore, can be ignored for this study.

3.3 Random-split in Source Model Training

In this study, we have trained our source models on a random 95%-5% train-validation split with early stopping. From our previous remarks about random-split, the credibility of these source models should also be questioned.

For our initial experiments, we were using the non-TL models we've trained for the random-split experiments as source models. When the problem of random-split arose, we've switched our source models to use 100% of the source data. This resulted in worse evaluations for the target model, due to the source network not being optimized for such a task. In an effort to minimize the negative effects of data leakage, while keeping the benefits of early stopping; we've decided on the 95%-5% random split. It should be noted that the 95%-5% split still negatively affects the training of the source models, and for a state-of-the-art product; source models should be first optimized on a compound-based split and trained on 100% of the source data.

3.4 Splitting Up and Down Regulations

In L1000, gene perturbations can be split into 3 categories; up-regulated, neutral, and down-regulated. Multi-class classification is a special case of classification problems and is usually harder to model compared to binary classification problems.

DeepCOP and subsequently this study divides what normally is a multi-class classification problem into 2 binary classifications. While this widely-used practice makes the data easier to model, it also comes at a cost of lost knowledge.

Specifically, in this application, the relation between up and down-regulated outputs is lost, because our binary classifications are "actively up-regulated to not significantly up-regulated" and "actively down-regulated to not significantly down-regulated". Training an ideal for the first problem would require a greater penalty to classify an actively down-regulated sample as actively up-regulated than to classify a non-regulated sample as one; and vice versa for the second problem.

Modeling the problem as multi-class showed that the features are not descriptive enough to model the added difficulty, at least for this type of neural network.

The most obvious solution would be using a custom loss function that uses the results of an oppositely-regulated problem. Another possible solution could be transfer learning. Utilizing the weights of up-regulated models in down-regulated models, or modeling up and down-regulation as subnetworks of the same neural network, has the potential to utilize the lost knowledge.

4 Conclusion

In this study, we tried to exhibit the benefit of using the natural genetic similarities in between cell lines for chemogenomic machine learning. While doing so, we have introduced 2 splitting methods to DeepCOP to better evaluate the strength of its methodology. Based upon the initial study of DeepCOP, our results have shown improvements of 10.65% on average for the introduced cold-drug-split. While there is a definite improvement in the results, this study should be treated as a simple demonstration, and can be further optimized for even better results. As it stands, this study is a testament for the usage of transfer learning in chemogenomics, and a demonstration on how it can be utilized. Moving forward, we're planning on implementing other methods of transfer learning, applying transfer learning on more than 2 cell line combinations, and optimize them to further illustrate the optimum possible result out of this methodology.

References

1. Bredel, M., Jacoby, E.: Chemogenomics: an emerging strategy for rapid target and drug discovery. Nature Reviews Genetics **5**(4), 262–275 (2004)
2. Consortium, G.O.: The gene ontology (go) database and informatics resource. Nucleic acids research **32**(suppl_1), D258–D261 (2004)
3. Gobbi, A., Poppinger, D.: Genetic optimization of combinatorial libraries. Biotechnology and bioengineering **61**(1), 47–54 (1998)
4. Li, H., Tian, S., Li, Y., Fang, Q., Tan, R., Pan, Y., Huang, C., Xu, Y., Gao, X.: Modern deep learning in bioinformatics. Journal of molecular cell biology **12**(11), 823–827 (2020)
5. Li, Y., Huang, C., Ding, L., Li, Z., Pan, Y., Gao, X.: Deep learning in bioinformatics: Introduction, application, and perspective in the big data era. Methods **166**, 4–21 (2019)
6. RDKit: Open-source cheminformatics. http://www.rdkit.org, [Online; Release 2021.3]
7. Rogers, D., Hahn, M.: Extended-connectivity fingerprints. Journal of chemical information and modeling **50**(5), 742–754 (2010)
8. Subramanian, A., Narayan, R., Corsello, S.M., Peck, D.D., Natoli, T.E., Lu, X., Gould, J., Davis, J.F., Tubelli, A.A., Asiedu, J.K., et al.: A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. Cell **171**(6), 1437–1452 (2017)
9. Tang, B., Pan, Z., Yin, K., Khateeb, A.: Recent advances of deep learning in bioinformatics and computational biology. Frontiers in genetics **10**, 214 (2019)
10. Woo, G., Fernandez, M., Hsing, M., Lack, N.A., Cavga, A.D., Cherkasov, A.: Deepcop: deep learning-based approach to predict gene regulating effects of small molecules. Bioinformatics **36**(3), 813–818 (2020)