Transferability of counterfactual examples across different models

Project Work - Responsible AI (by Bilal Zafar)

Frederik Hüttemann
Applied Computer Science
Ruhr-Universität Bochum
`frederik.huettemann@rub.de`

Abstract

An abstract should concisely (less than 300 words) motivate the problem, describe your aims, describe your contribution, and highlight your main finding(s). // In this project, we investigate the transferability of counterfactual examples across different models. Neural Networks are often used to make life critical decisions, such as in healthcare or criminal justice. They also find applications in less critical domains, such as image classification. However, these models are often criticized for being black boxes, which makes it difficult to understand their decisions. Counterfactual examples are a method to explain the decisions of a model by showing how the input would have to change in order to change the output.
It is often assumed that counterfactual examples are model-specific, so their the same counterfacutal example usually cannot be applied on different models. The goal of this project is to look deeper into this assumption and try to find out whether counterfactual examples can be transferred across different models. For this, different types of data and models are used. For the simple cases two numerical datasets are used. For the more complex case, the Imagenet dataset is converted into a binary classification task.
TBD: results and conclusion The results show that counterfactual examples can be transferred across different models, but the transferability depends on the type of data and the model used. In particular, counterfactual examples for numerical data can be transferred to at least some extent, while counterfactual examples for image data are not transferable at all. This suggests that the transferability of counterfactual examples is not a general property, but rather depends on the specific data and model used.

1 Introduction

Counterfactual examples are a method to explain a model's decisions by showing how one or more input features would have to change in order to change the output of the model. This explainability method is often used in life-critical domains, such as healthcare or criminal justice, where a binary classification task is performed. For example, in criminal justice, a model might predict whether a defendant will reoffend within two years.
In this project, it is investigated whether counterfactual examples for the base model also work for different models or if they are model-specific. This is done by gradually increasing the difference between the base model and the reference model. In the first experiements both model only differ in the number of trained epochs. In the other experiements the models differ in the number of layers, features per layer, and the architecture. The goal is to find trends in the transferability of counterfactual examples across different models.
For the experiements two different types of data is used: numerical data and image data. This is done because for numerical data it is expected that the transferability of counterfactual examples is higher than for image data. This is because numerical data is more structured and therefore easier to

understand for the model. For image data, model might focus on different features and therefore learn different decision boundaries.

2 Related Work

Counterfactual examples are a well-known method to explain the decisions of a model. Machine learning is used in many different domains, such as healthcare, job applications and administrative decisions. In these domains, it is often important to understand the model's decisions and to be able to explain them. Using counterfactual examples, a user can understand how the model's decision would change if one or more input features were changed. This is especially important in life-critical domains, where wrong decisions can have severe consequences. [4]
There are approaches to generate more general, diverse and feasible counterfactual examples. A set of them can be used to understand local model behavior and approximate the decision boundary at a given point.[4]
Additionally, it is stated that the closely linked adversarial examples can be used to attack a model by generating inputs that are misclassified by the model. This is done by changing the input features in a way that the model's decision changes. Adversarial examples can that mislead the base model can also mislead other models which can be different in architecture, training data, and hyperparameters. In the paper [5] an approach is developed which allows attacks on unknown models by using adversarial examples generated on a known model. They managed to generated general adversarial examples that lead to a high number of misclassifications. [5]

3 Approach

In the related work section, it is stated that counterfactual examples are often model-specific and therefore cannot be transferred across different models. However, it is also known that adversarial examples, which are are similar to counterfactual examples, can be used to attack unknown different models.
In this project, an exploratory approach is taken. The goal is to find a limit on how far counterfactual examples can be transferred. This means that the project does not focus on a specific method, but rather on the general idea of counterfactual examples and their transferability.
In general, every experiment performed in this project uses two models. The first model is called the *base model* and is used to generate counterfactual examples. The second model is called the *reference model* and is used to evaluate the transferability of the counterfactual examples. Both models are trained on the same training data to ensure that the counterfactual examples are not biased towards a specific model.

4 Experiments

The experiments, which are described in this section, are designed to test the transferability of counterfactual examples across different models. We fist of all consider two different types of models: simple feed-forward neural networks and more complex convolutional neural networks. Feed-forward neural networks are more simple and therefore more promising for the transferability while the second type of model is more complex and therefore unlikely to reliably transfer counterfactual examples.
To evaluate the transferability of counterfactual examples, a metric is needed that quantifies how well the counterfactual examples transfer to the reference model. This metric is called the *transferability rate*. The transferability rate is given by equation 1 and is defined as the ratio of the number of counterfactual examples that are correctly classified in the counterfactual class by the reference model to the total number of counterfactual examples generated by the base model. The transferability rate is a value between 0 and 1, where 0 means that none of the counterfactual examples are correctly classified in the counterfactual class by the reference model and 1 means that all counterfactual examples are correctly classified in the counterfactual class.

$$\text{transferability rate} = \frac{\text{counterfactual examples classified in counterfactual class}}{\text{total number of counterfactual examples}} \tag{1}$$

Because the training and generation of counterfactual examples is computationally expensive, the transferability rate is only calculated for a limited number of counterfactual examples. Additionally, the generation of counterfactual examples is limited to a maximum of 1000 iteration. This means that for some datapoints the counterfactual example cannot be generated within the given number of iterations. In this case, the counterfactual example is not counted towards the total number of counterfactual examples. This means that the transferability rate is not a perfect measure of the transferability of counterfactual examples, but it is a good approximation.
Multiple experiments are performed to test the transferability. To start simple, the first experiments are performed on numerical, tabular data and using simple feed-forward neural networks. The performed experiments are:

- **Influence of continued training:** The first experiment investigates how the transferability is changed when the reference model is the same model as the base model, but trained for more epochs. The base model is trained for 10 epochs. The reference model is a copy of the base model and its weights but trained for 20 additional epochs. For each new epoch, the *transferability rate* is calculated. The assumption is that the transferability rate might decreases because for every new epoch the reference model can learn more and different features of the data, which makes it less likely that the counterfactual examples generated by the base model are still correctly classified.
- **Influence of model parameters:** In this experiment, the base model and reference model are the same model, so the architecture, number of layers and number of neurons per layer is identical. Additionally, both models are trained for the same number of epochs. Because the models are the same, the only difference is the random initialization of the weights. For more complex models, the assumption is that the transferability rate might be lower because the model can learn more complex features and decision boundaries and therefore might not be able to classify the counterfactual examples correctly.
- **Influence of model more layers:** Here, the base model has a different architecture than the reference model. Given a base model with a certain number of layers, the reference model has additional layers. The number of features per layer is fixed and the same for both models. The goal of this experiment is to find out whether the transferability rate is related to the number of layers in the model.
- **Influence of model more features:** In this experiment, the base model has again a different architecture than the reference model. Given a base model with a certain number of features per layer, the reference model has more features per layer. The number of layers is fixed and the same for both models. The goal of this experiment is to find out whether the transferability rate is related to the number of features in the model. One can assume that the transferability rate is lower for models with more features because the learned decision boundaries are more complex.

The next experiments are performed on image data and using convolutional neural networks. Here, the transferability rate is calculated for the same models as in the previous experiments, but with a different architecture. The performed experiments are:

- **Influence of continued training:** The first experiment is the same as for the numerical data. The reference model is a copy of the base model and its weights. For each additional epoch, the transferability rate is calculated.
- **Influence of model parameters:** Similar to the experiement for numerical data, the base model and reference model are the same model. Both have the same architecture, number of layers, and number of neurons per layer. The difference is that both models are convolutional neural networks. Therefore the number of parameters is much higher than for simple feed-forward neural networks. The Convolution Kernels are initialized randomly, so they often learn different features. Therefore, it is expected that the transferability rate is much lower than for numerical data.

Because each experiment is very time consuming, only a limited number of experiments are performed. Also a low number of iterations is used so the results might suffer from noise. Therefore the results can be interpreted as a trend and not as a final result.

4.1 Data

The used data can be divided into two categories: numerical data and image data. For the numerical data, two datasets are used: the *Compas* dataset [6] and the *ACS* dataset [1].
The *Compas* dataset contains information about criminal defendants, such as their age, the prior number of offenses, and the type of offense. The goal is to predict whether a defendant will reoffend within two years.
The *ACS* dataset contains information about the American Community Survey, such as the age, education, and income of individuals. The dataset label is wheather the the data point represents a person who is employed or not.
For image classification data there are many different possible datasets to choose from. As I wanted a binary classification task, I had to convert all image classification datasets into a binary classification task. The used datasets are *CIFAR-10* [3] and *Imagenet* [2].
For *CIFAR-10*, the images are low resolution (32x32 pixels) images of 10 classes. To convert it to a binary classification task, every datapoint representing an *airplane* becomes a positive example, while all other datapoints become negative examples. Additionally, the negative and positive examples are balanced, so that the number of positive and negative examples is equal.
The same approach has been taken for *Imagenet*, where the images are high resolution (224x224 pixels) images and representing 1000 classes. The positive class is *orange* and the negative class is everything else. Again, the positive and negative examples are balanced, so that the number of positive and negative examples is equal.
The datasets are split into a training set and a test set. The training set is the same for all experiments, the test set is on the one hand used to evaluate and compare models but also to generate counterfactual examples. This way, the counterfactual examples are not biased towards a specific model because they are generated on the same, new data.

4.2 Evaluation method

The evaluation method is based on the transferability rate, which is defined in equation 1. The transferability rate is calculated for each experiment and for each model. The transferability rate is a value between 0 and 1, where 0 means that none of the counterfactual examples are correctly classified in the counterfactual class by the reference model and 1 means that all counterfactual examples are correctly classified in the counterfactual class.
Additionally, for some experiements a linear function is fitted to show the trend of the transferability rate.

4.3 Results

Report the quantitative results that you have found. Use a table or plot to compare results and compare against baselines.

- If you're a default project team, you should **report the accuracy and Pearson correlation scores you obtained on the test leaderboard** in this section. You can also report dev set results if you'd like.
- Comment on your quantitative results. Are they what you expected? Better than you expected? Worse than you expected? Why do you think that is? What does that tell you about your approach?

5 Analysis

Your report should include *qualitative evaluation*. That is, try to understand your system (e.g., how it works, when it succeeds and when it fails) by inspecting key characteristics or outputs of your model.

6 Conclusion

Summarize the main findings of your project and what you have learned. Highlight your achievements, and note the primary limitations of your work. If you'd like, you can describe avenues for future work.

7 Outlook

Because of limited time, the experiments are not exhaustive and only a limited number of experiments are performed. More experiments could be performed especially on the image data as each experiment takes a long time to run. For the last three weeks my personal GPU did run non stop to generate counterfactuals and train the models. Eventough I started early with the experiments, I was not able to finish all experiments I wanted to perform.
The generation of counterfactual examples does not take into account that the feature values might have different values. For tabular data some features represent binary values while others represent continuous values. The counterfactual algorithm used does not take those value ranges into account, which should lead to lower results as it then only relies on the model's bias.
I also limited my experiments on neural networks while other decision making models could be used as well. Examples could be decision trees or a Kernel SVM. These models are often more interpretable as they generate linear decision boundaries. Here, it might also be possible to draw the decision boundaries as well as the counterfactual examples in a 2D plot. This would allow a visual understanding of why some counterfactuals are transferable and others are not. For this, apporaches like PCA or t-SNE could be used to reduce the dimensionality of the data.

References

[1] U.S. Census Bureau. American community survey (acs) dataset, 2018. Accessed: 2025-07-01.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[3] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[4] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 607–617, New York, NY, USA, 2020. Association for Computing Machinery.

[5] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples, 2016.

[6] ProPublica. Compas dataset, 2016. Accessed: 2025-07-01.

A Appendix (optional)

If you wish, you can include an appendix, which should be part of the main PDF, and does not count towards the 6-8 page limit. Appendices can be useful to supply extra details, examples, figures, results, visualizations, etc. that you couldn't fit into the main paper. However, your grader *does not* have to read your appendix, and you should assume that you will be graded based on the content of the main part of your paper only.