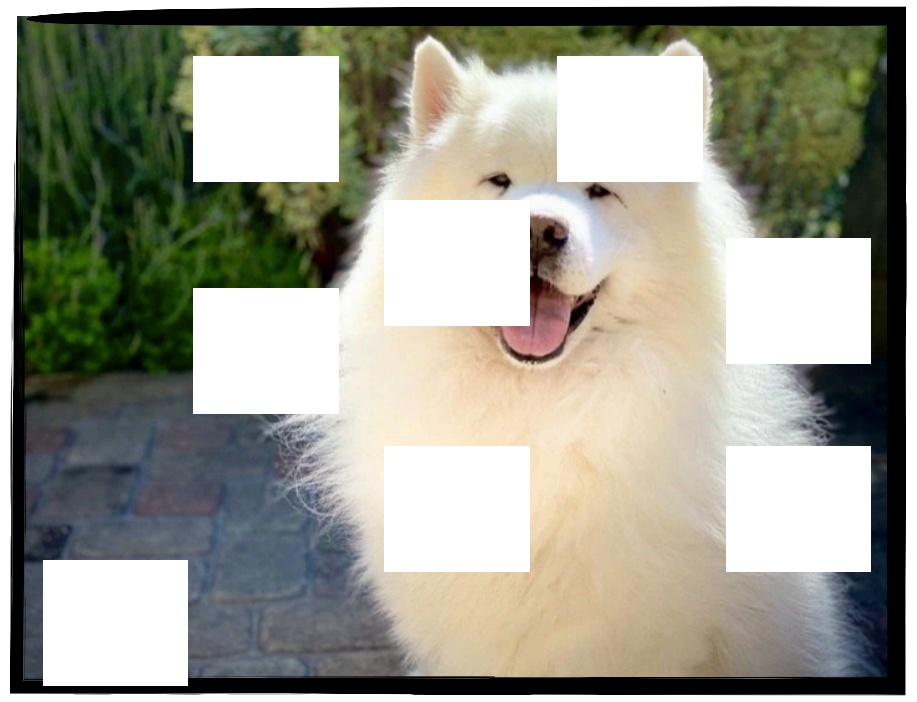
Self-Supervised Learning for Neutrino Experiments

NPML Tufts 2023

Radi Radev, Junior Fellow @ CERN

Self-Supervised Learning in Vision



- You have a lot of data but not many labelled examples
- Train some model that utilises the unlabelled data
- Then you can fine-tune the base model using the small labeled sample

Self-Supervised Learning in Vision

- You have a lot of data but not many labelled examples
- Train some model that utilises the unlabelled data
- Then you can fine-tune the base model using the small labeled sample

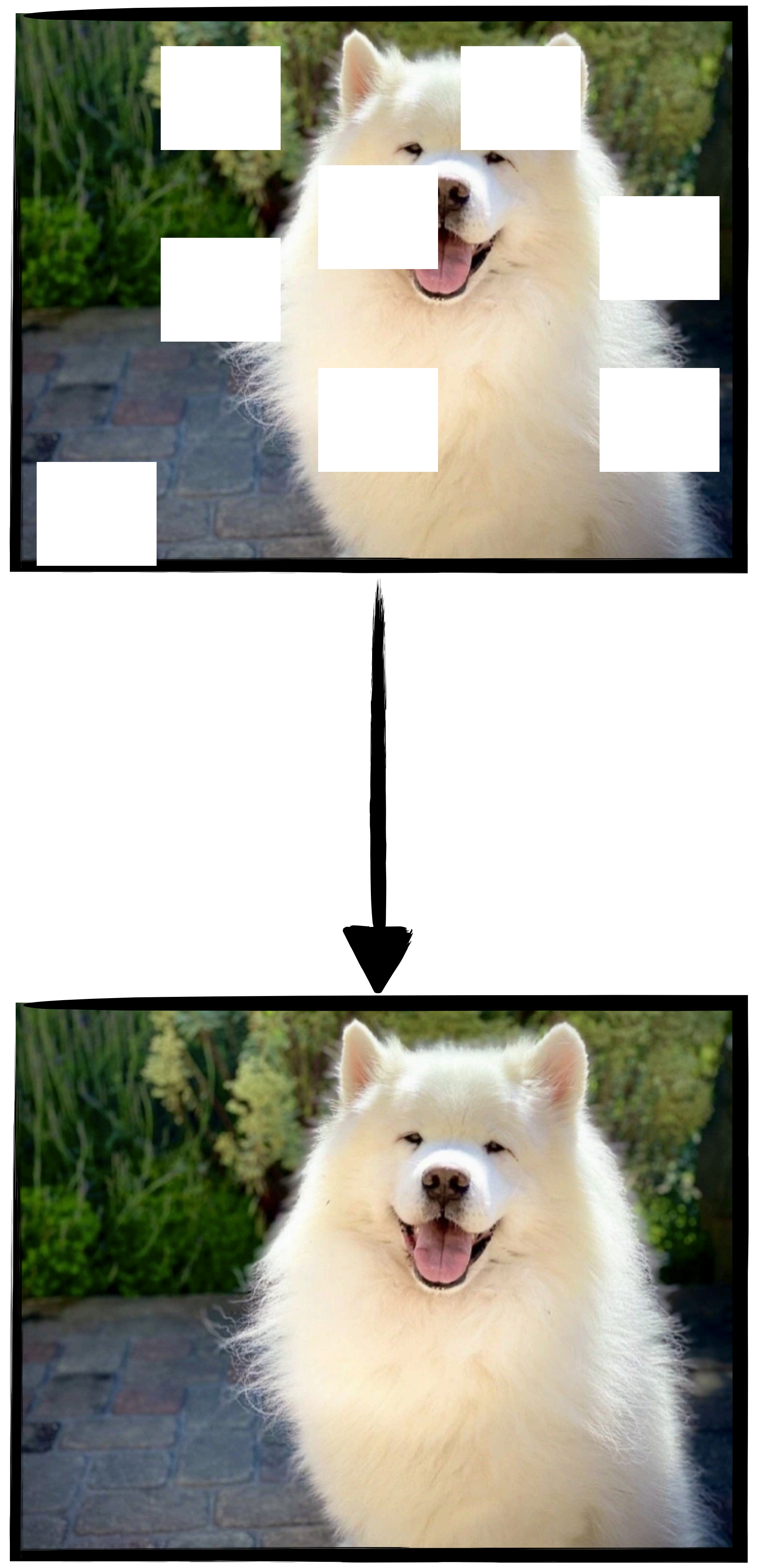


Illustration of MAE - vision foundation model

Self-Supervised Learning in Vision

- You have a lot of data but not many labelled examples
- Train some model that utilises the unlabelled data
- Then you can fine-tune the base model using the small labeled sample
- But HEP simulation comes with detailed information?
- It can help mitigate **biases** we have in our simulation

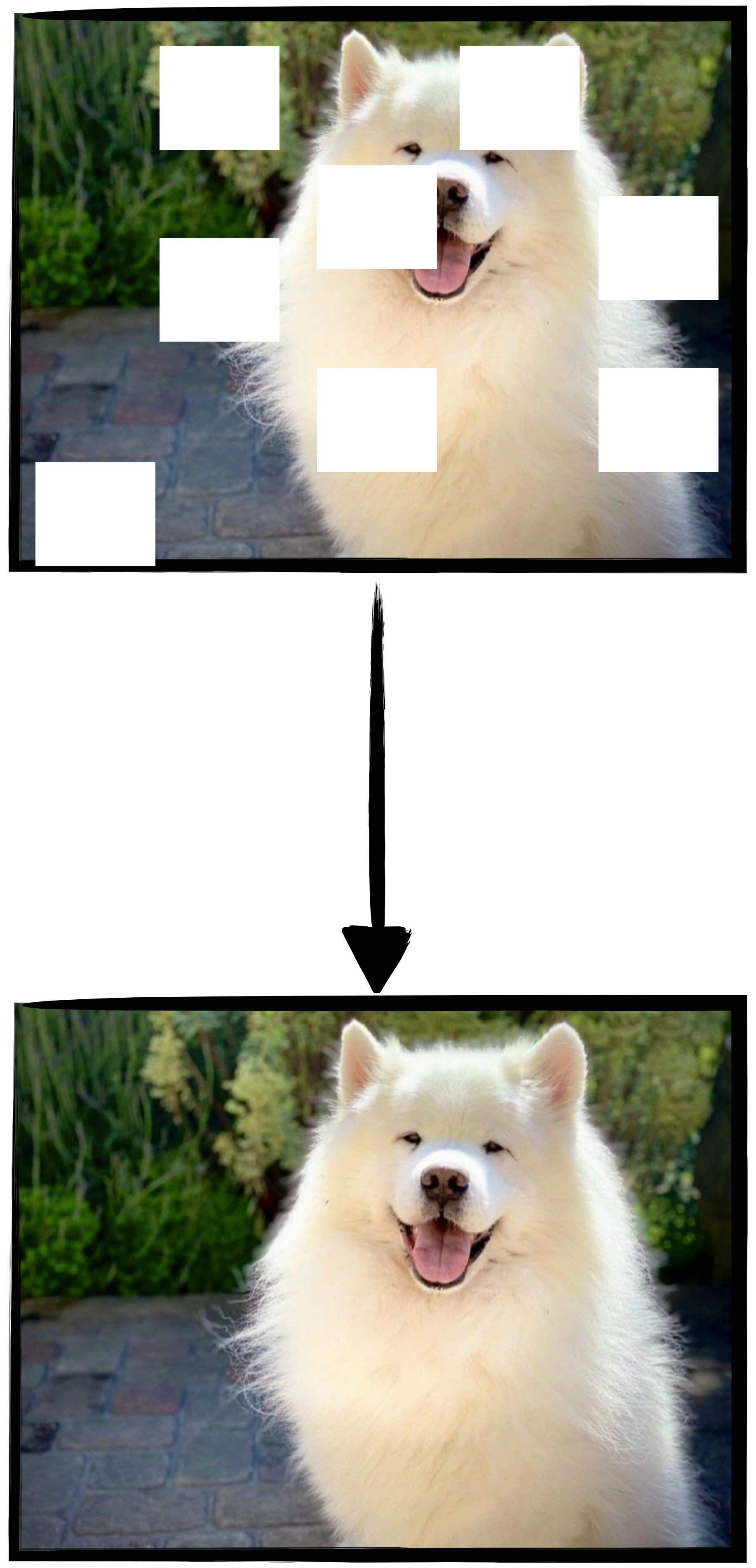


Illustration of MAE - vision foundation model

Mitigating biases by Pretraining

Self-supervised methods do not require labeled data.

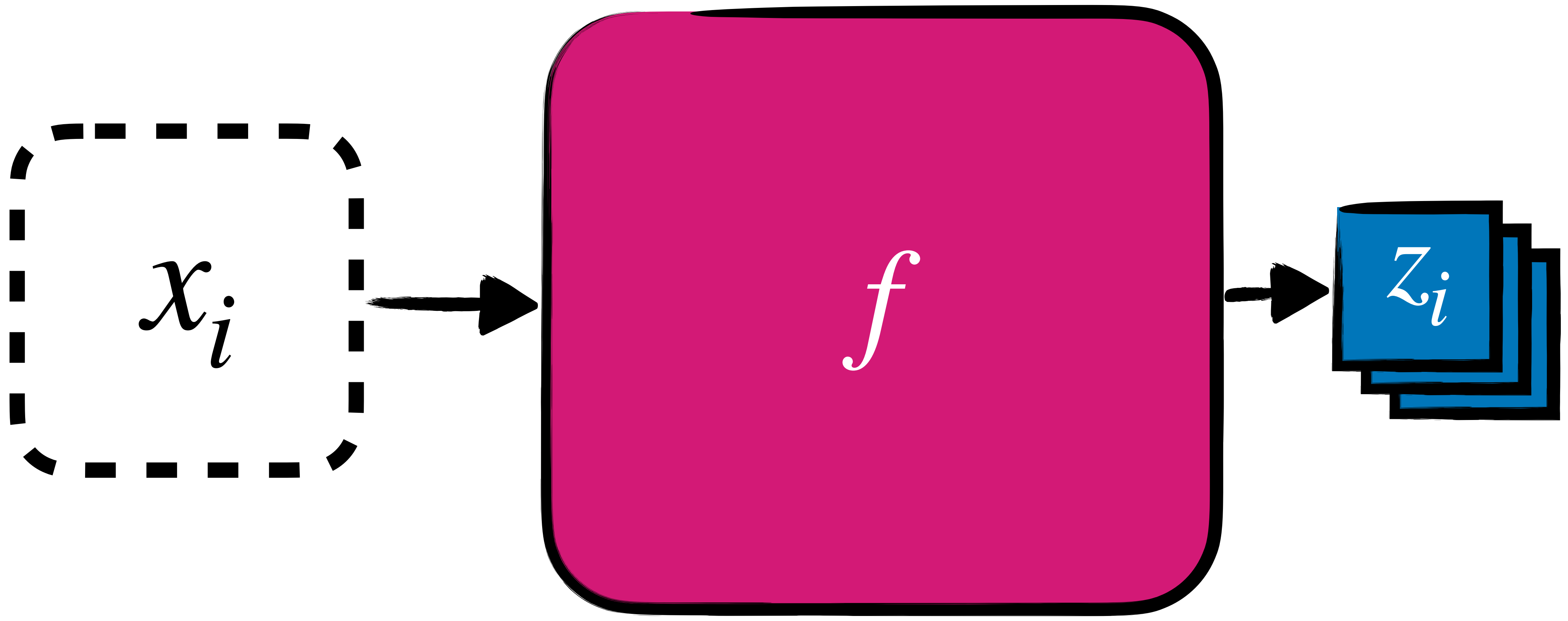
Can be trained directly on real data only or a combination of Monte-Carlo and real data

Furthermore we can use our detector systematic shifts in the pretraining phase - making the model invariant to variations in it.

SimCLR naturally adresses all the above points

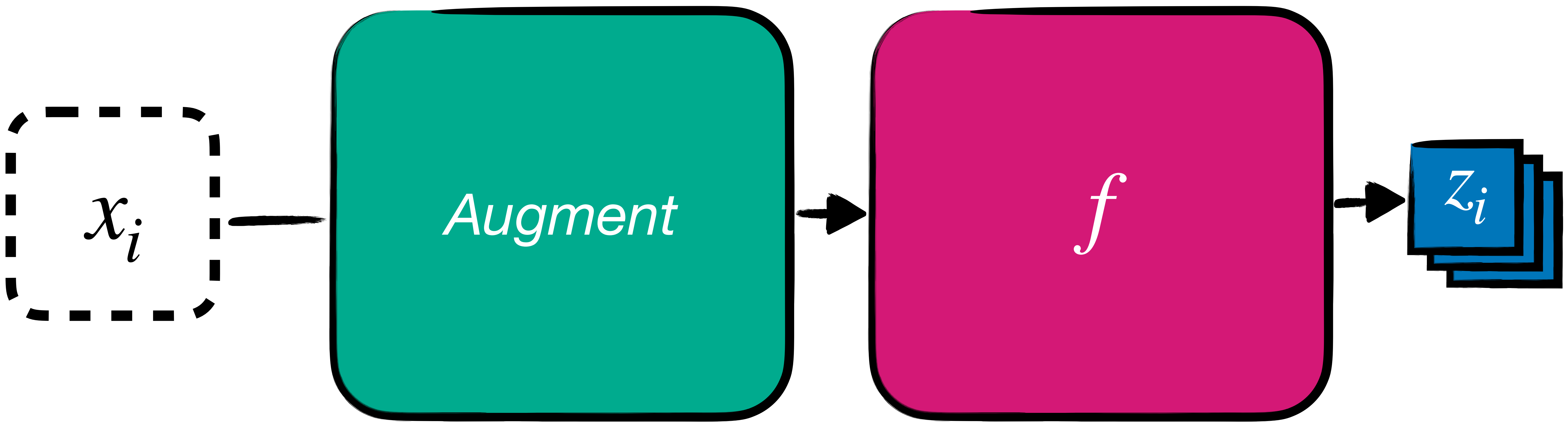
SimCLR Overview

Representations



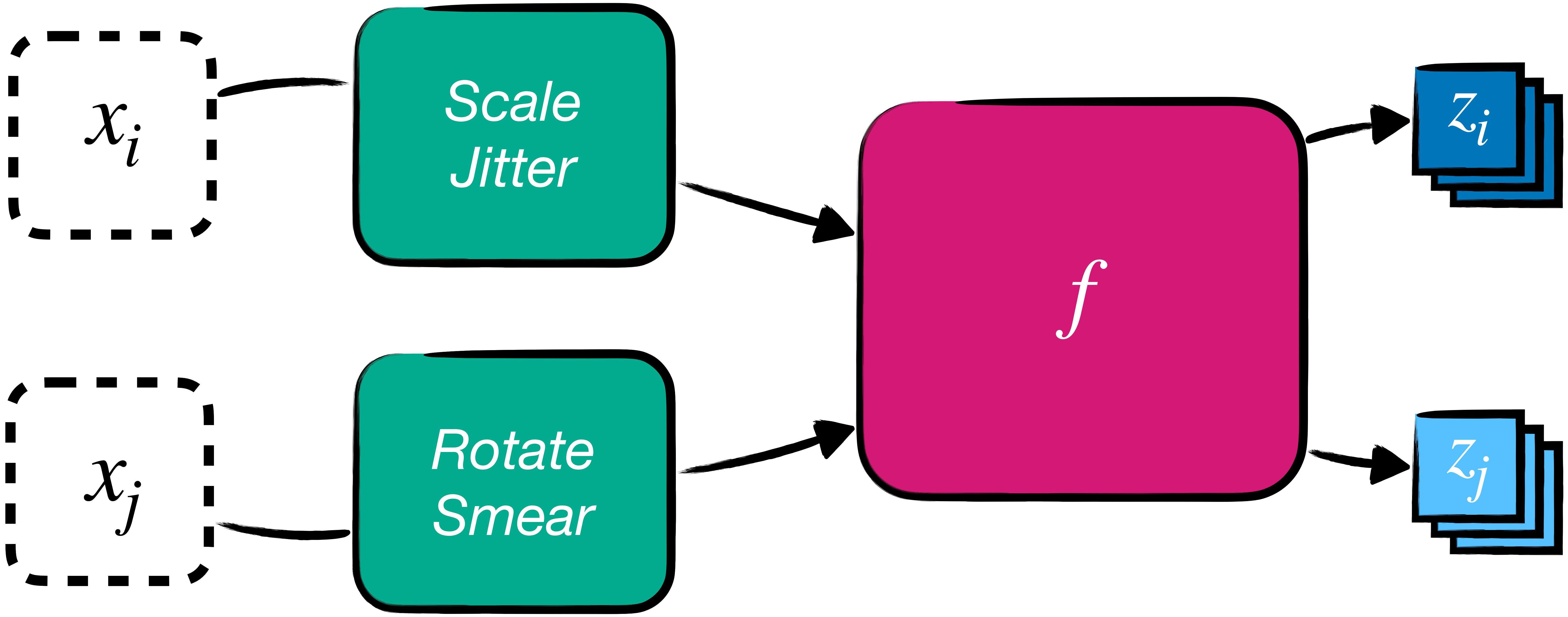
Pass an **event** x_i through a **neural network** f to extract a **vector representation** z_i.

Representations



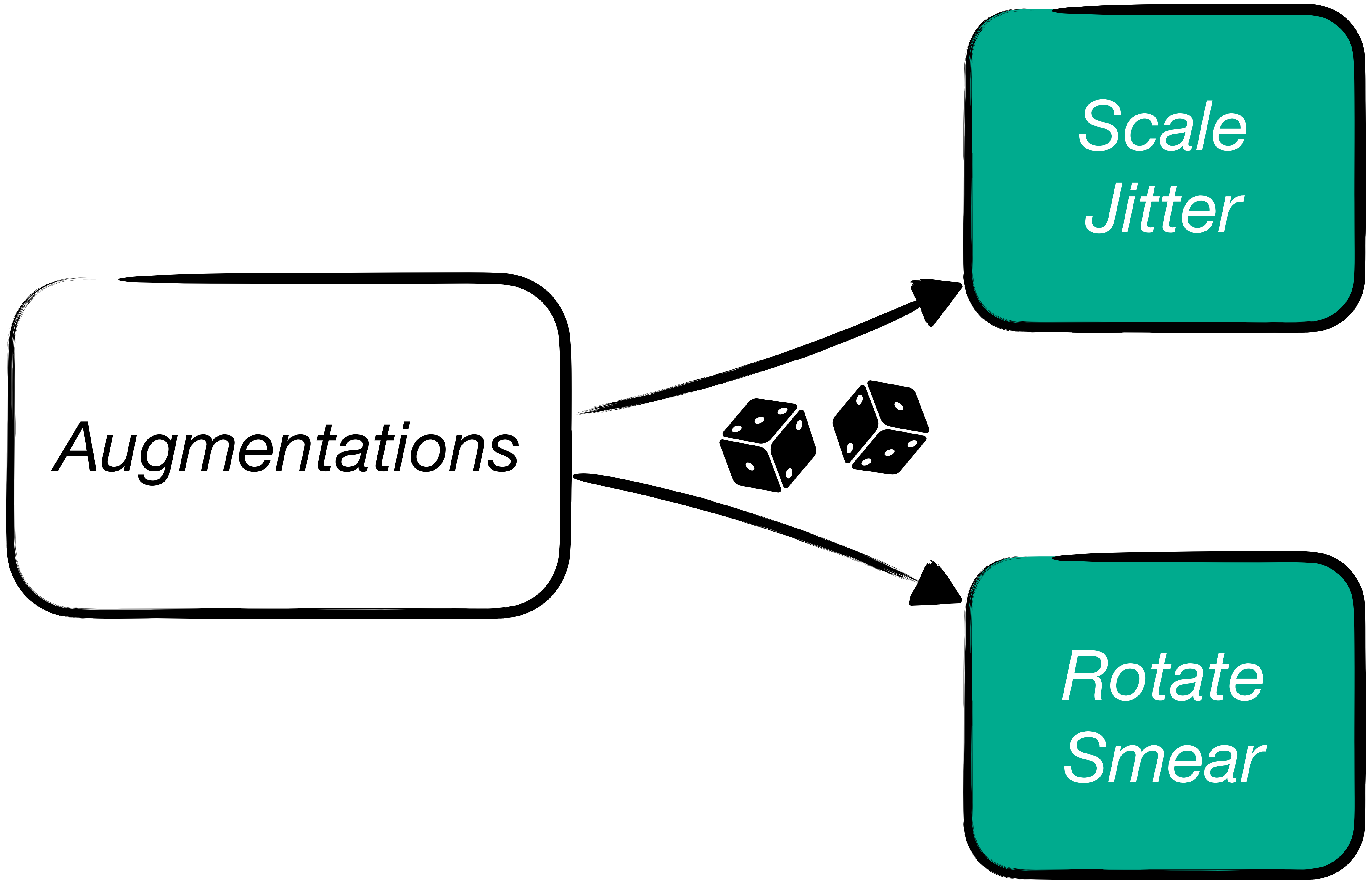
Pass an **augmented** **event** x_i through a **neural network** f to extract a different **vector representation** z_i.

Contrastive Learning



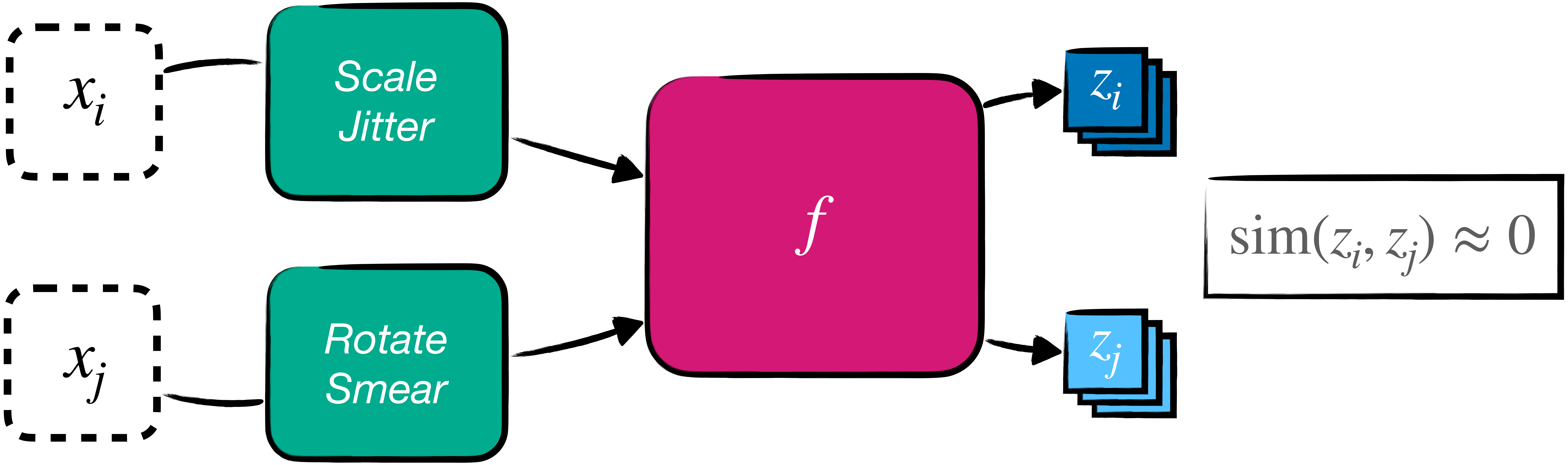
Pass pairs of **augmented events** through a **neural network** f to extract **vector representations.**

Contrastive Learning



In practice the set of **augmentations** to be applied to the pairs is picked randomly for each training iteration.

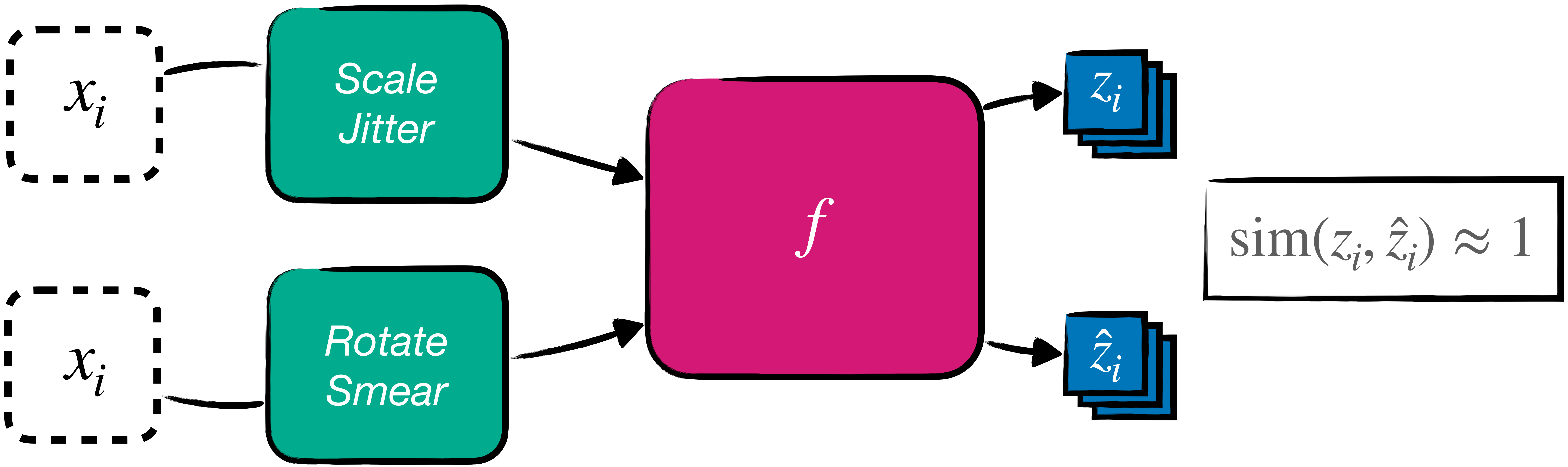
Contrastive Learning



Pass pairs of **augmented** **events** through a **neural network** f to extract **vector representations.**

Representations from **different** events - **low similarity**

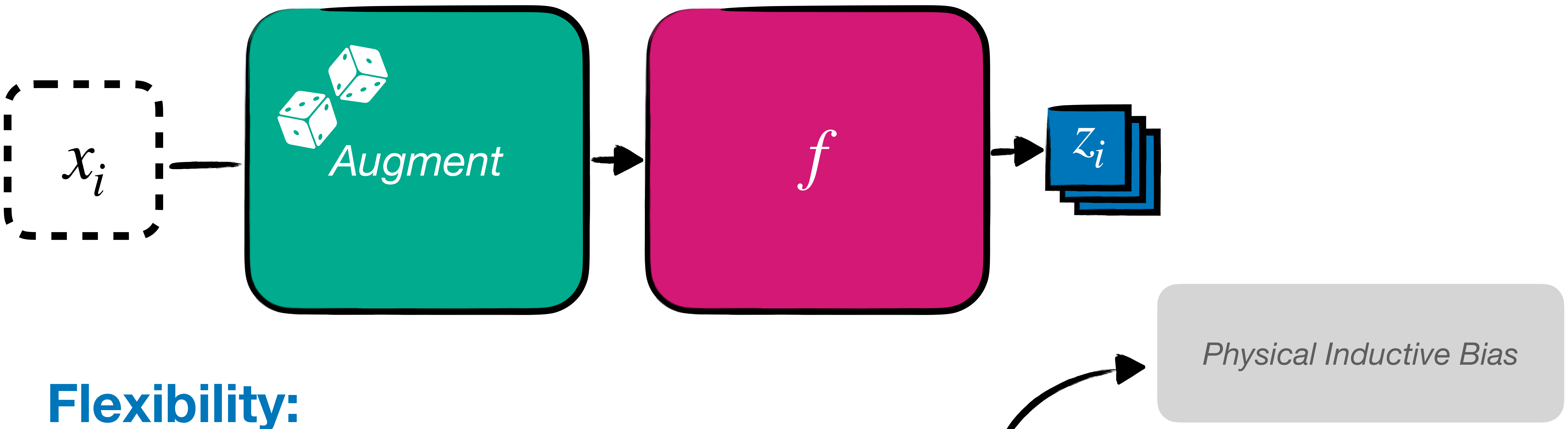
Contrastive Learning



Pass pairs of **augmented** **events** through a **neural network** f to extract **vector representations.**

Representations from **same** event - **high similarity**

Contrastive Learning

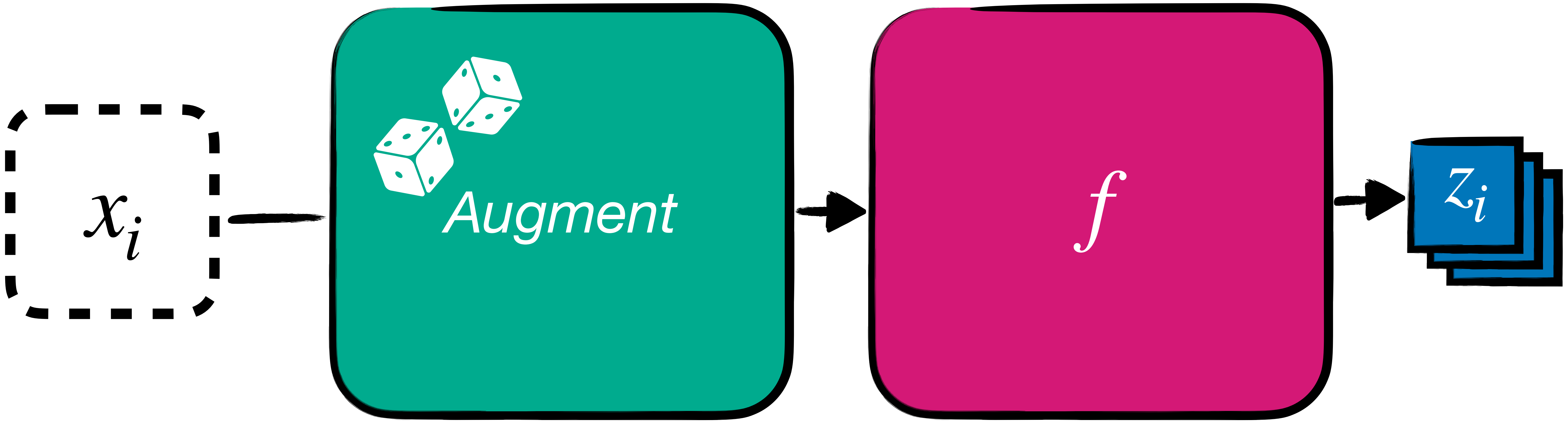


Flexibility:

Use any **augmentation** - What invariance do we encode?

Use any **neural network** - What is the most natural data structure of the event?

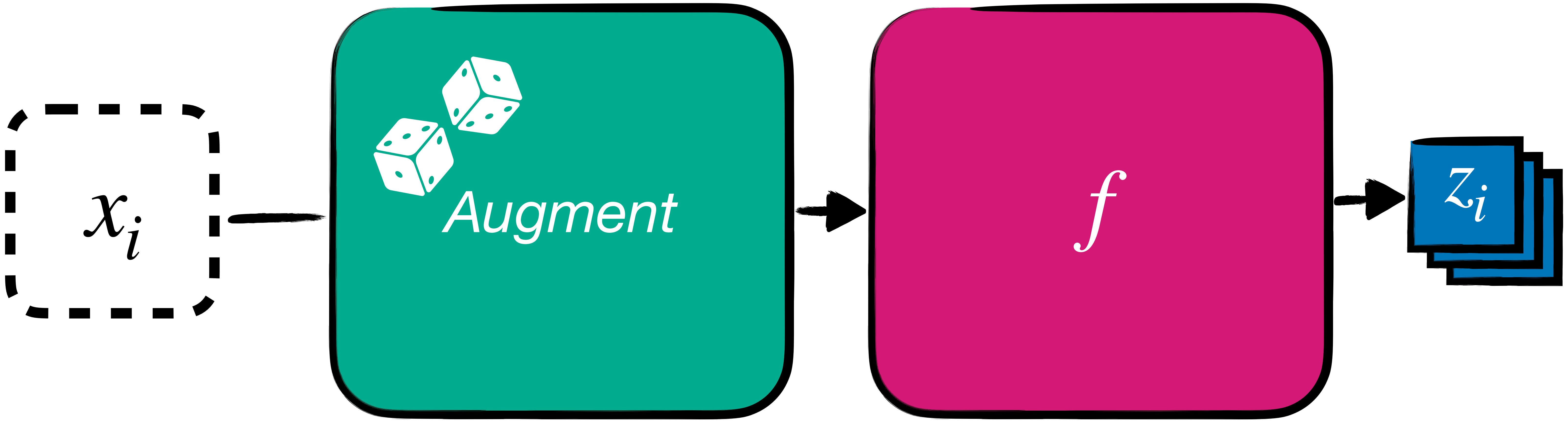
Contrastive Learning



No labels needed - can pre-train on **real** data!

Dataset and Method

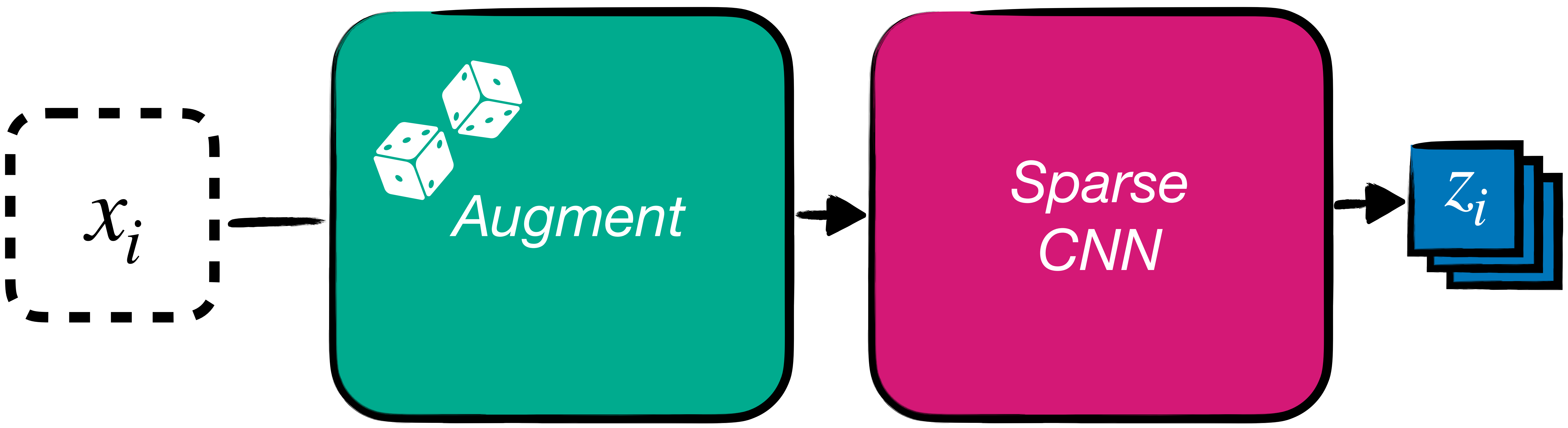
Method



Augmentations:

- random scaling, translation, rotation, dropping voxels

Method



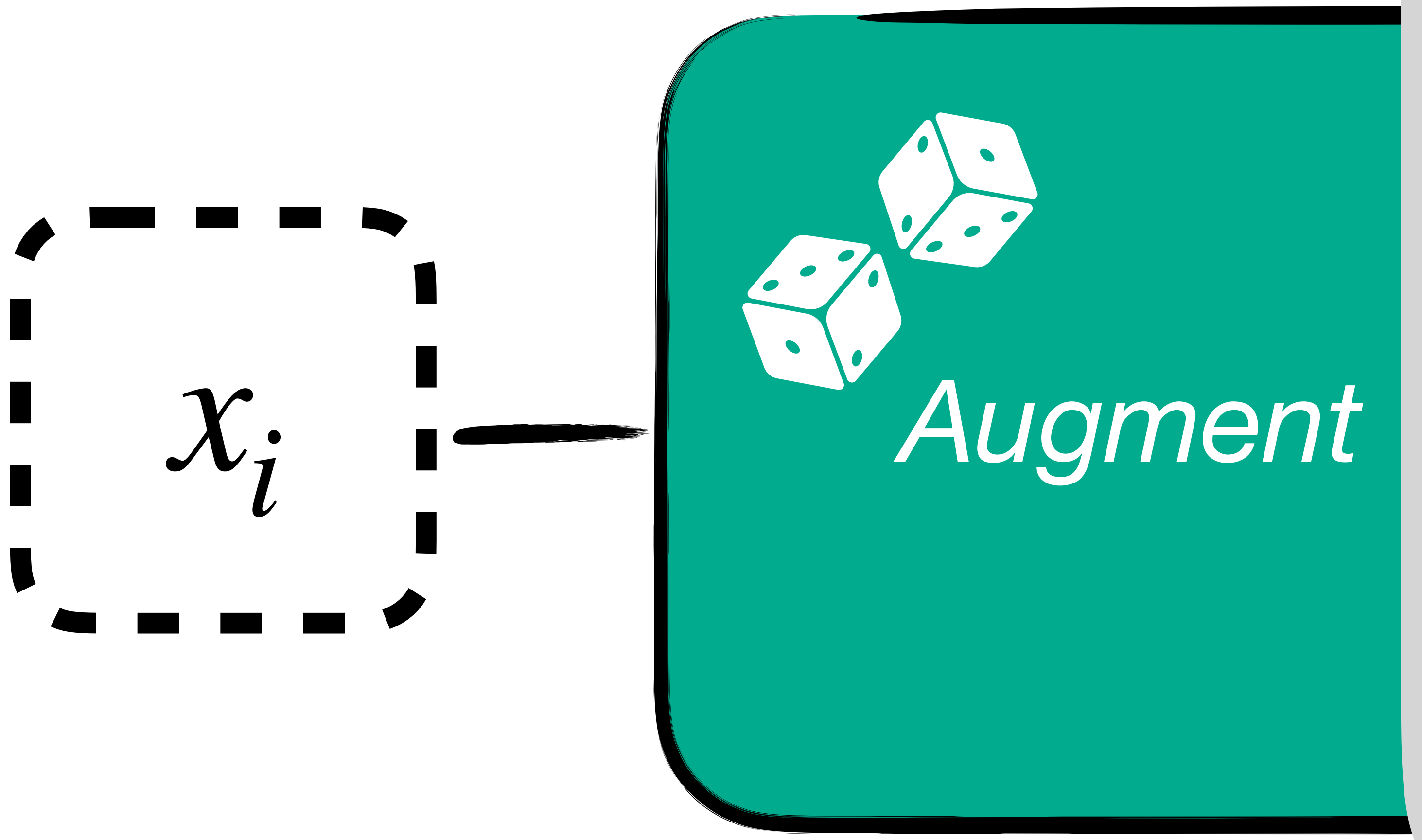
Augmentations:

- random scaling, translation, rotation, dropping voxels

Architecture:

- a sparse sub manifold CNN based on ConvNeXt v2

Method



But wait aren't CNNs already invariant to translations?

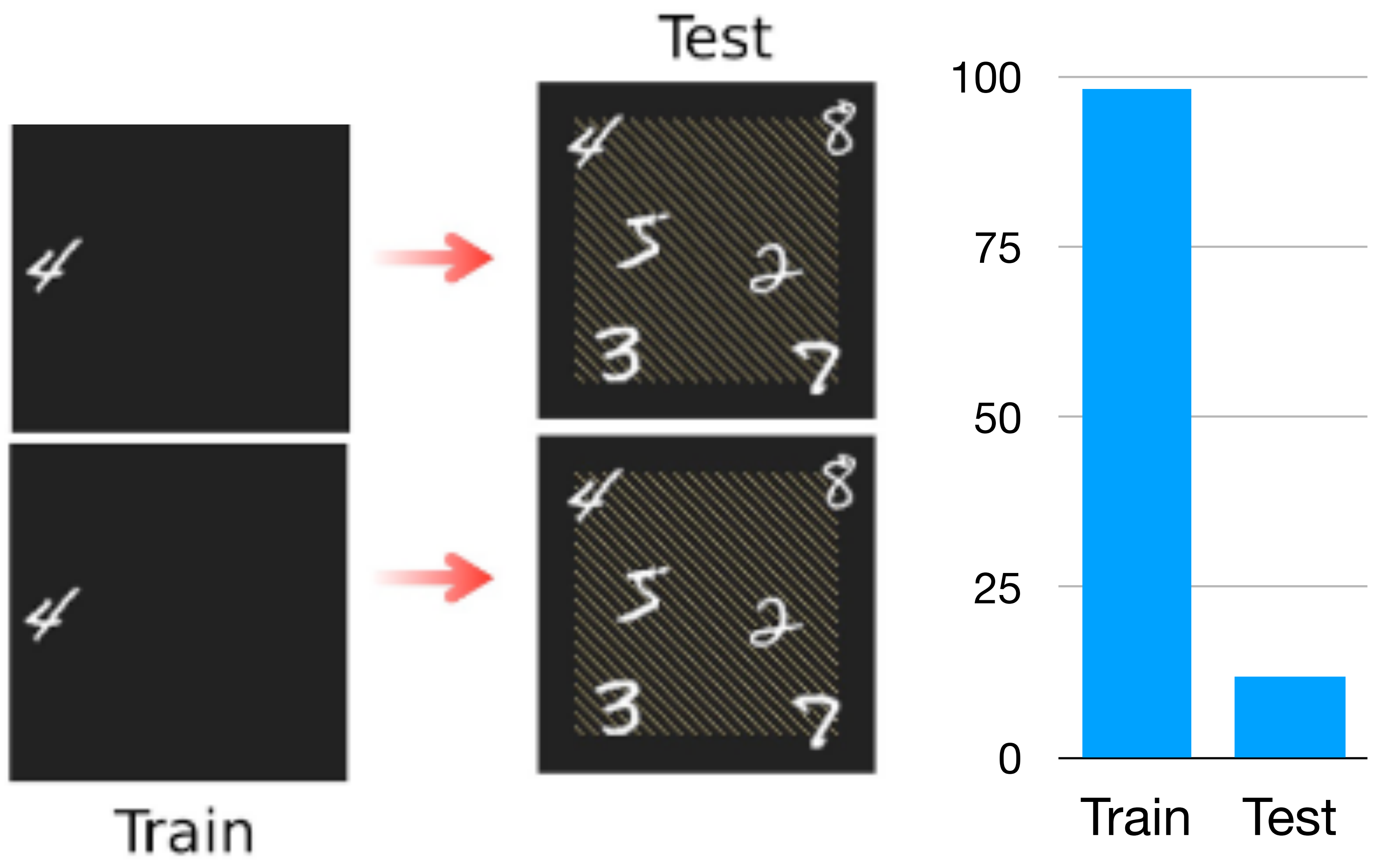
Augmentations:

- random scaling, translation, rotation, dropping voxels

Architecture:

- a sparse sub manifold CNN based on ConvNeXt v2

Aside - CNN Translation Invariance



Adapted From "CNNs Are Not Invariant to Translation, but They Can Learn to Be"

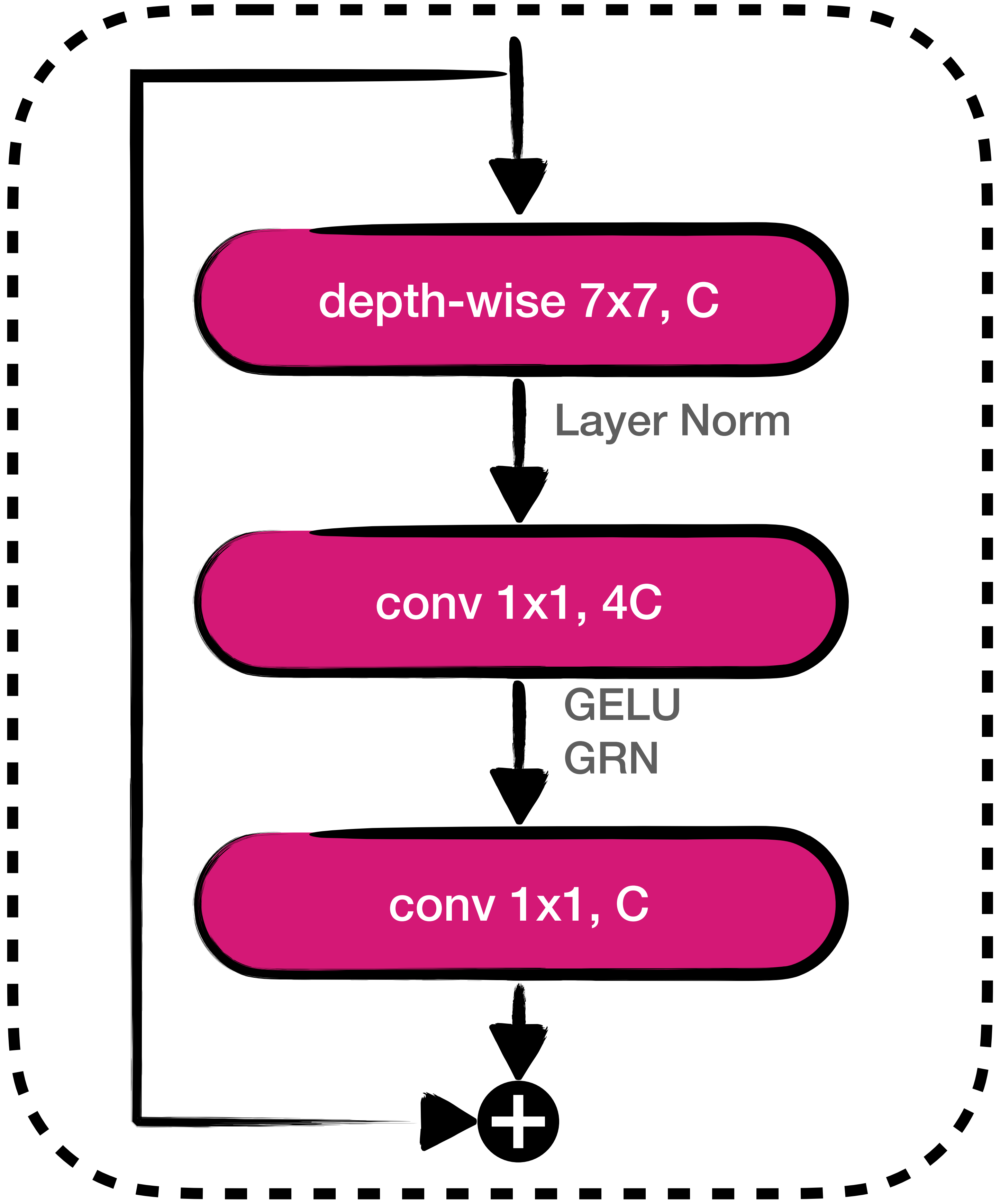
Turns out not quite!

But wait aren't CNNs already invariant to translations?

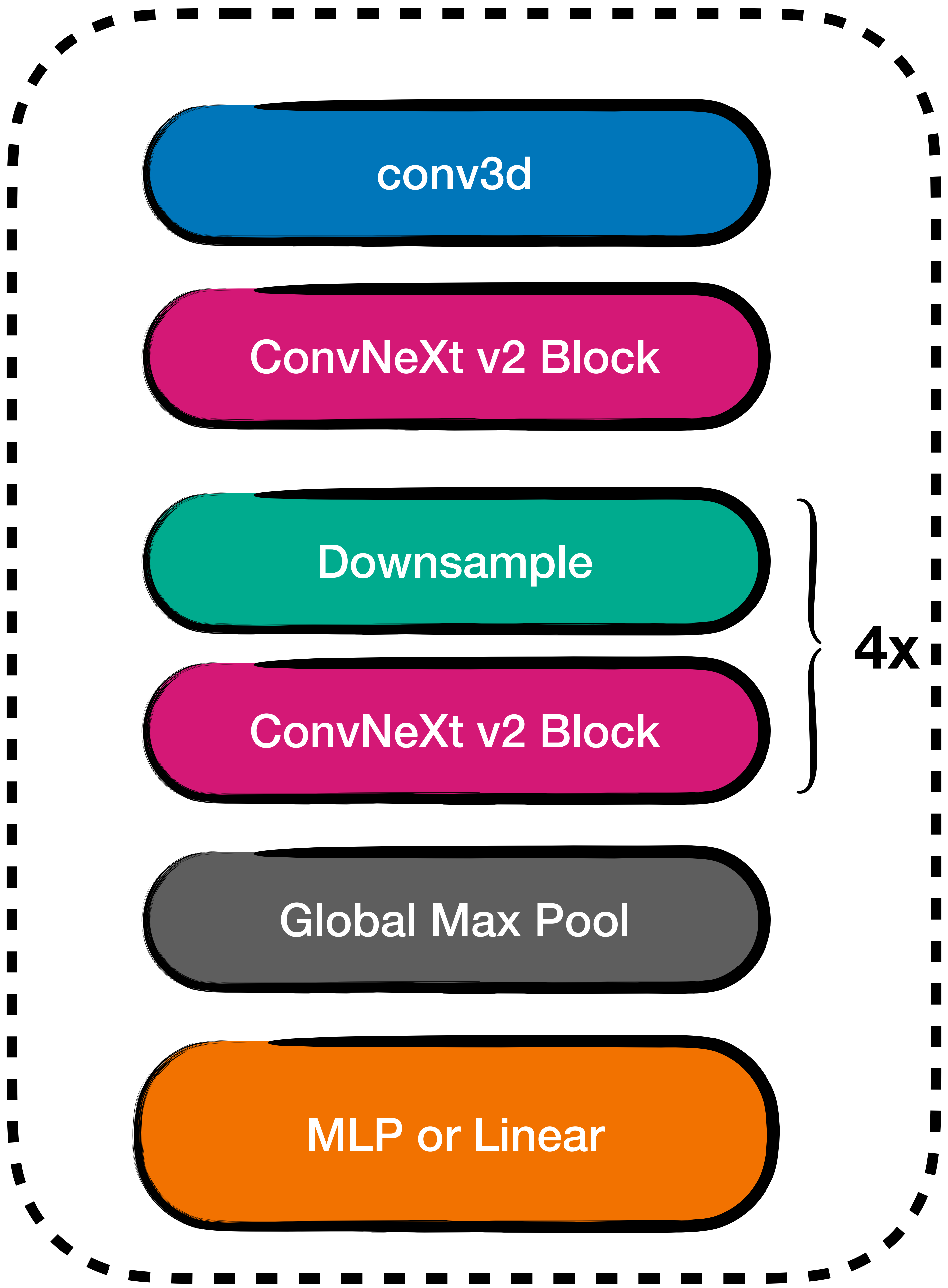
Convolutions are **equivariant** to translation, but this does not directly translate to invariance.

Although architectures can be constructed to be invariant to translations, most modern CNNs are not by default

Architecture



ConvNeXt v2 Block



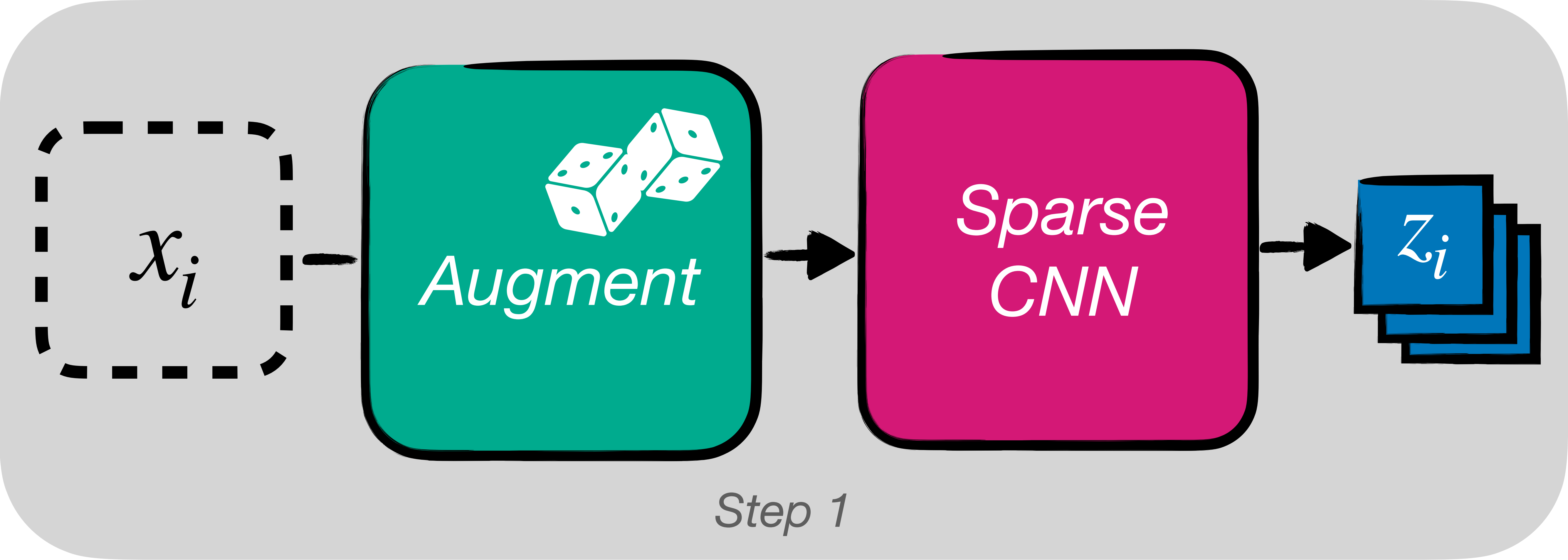
Full architecture

We use an MLP to get the similarity vector for CLR and a Linear layer if we are training a classifier.

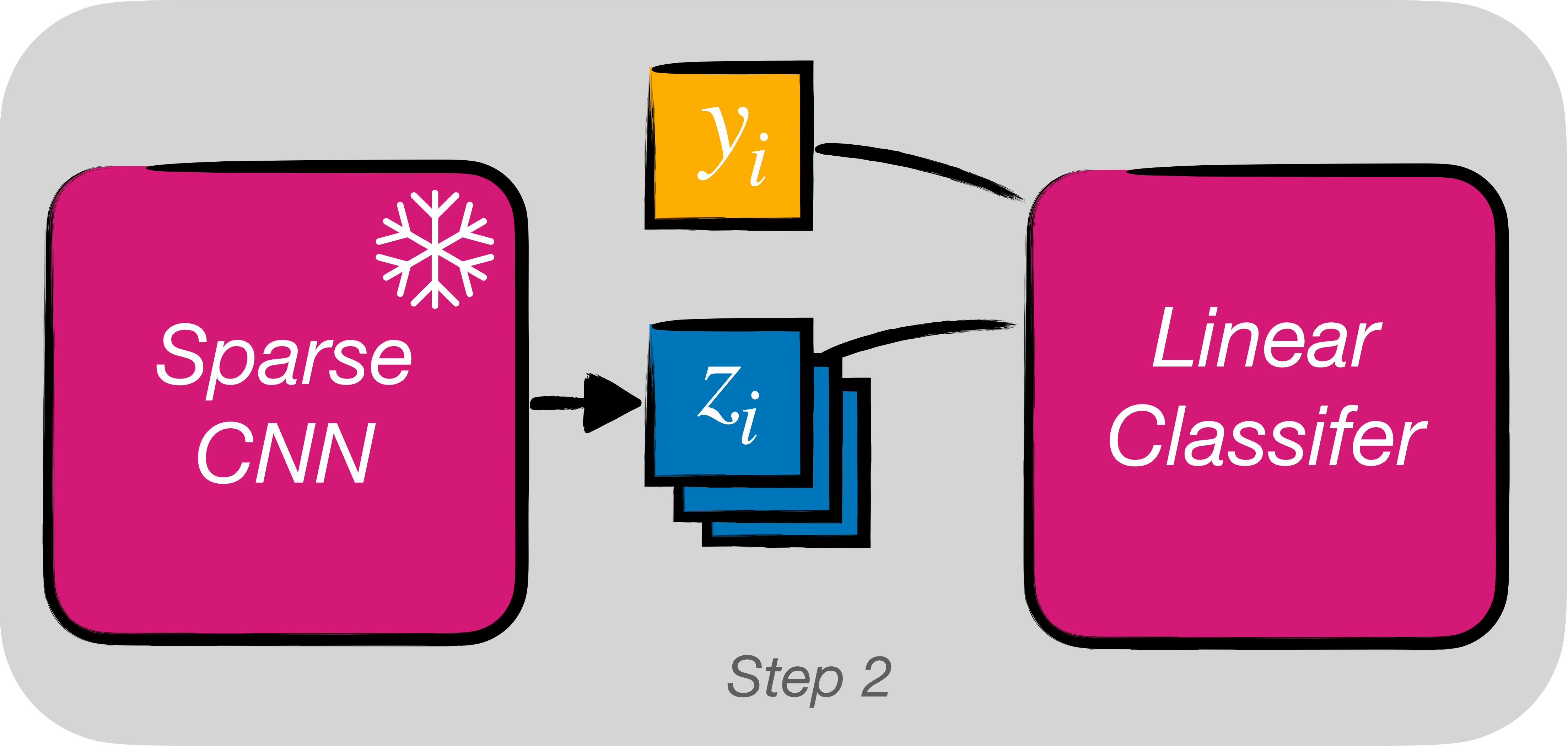
Architecture:

- a sparse submanifold CNN based on ConvNeXt v2

Training and Evaluating SimCLR



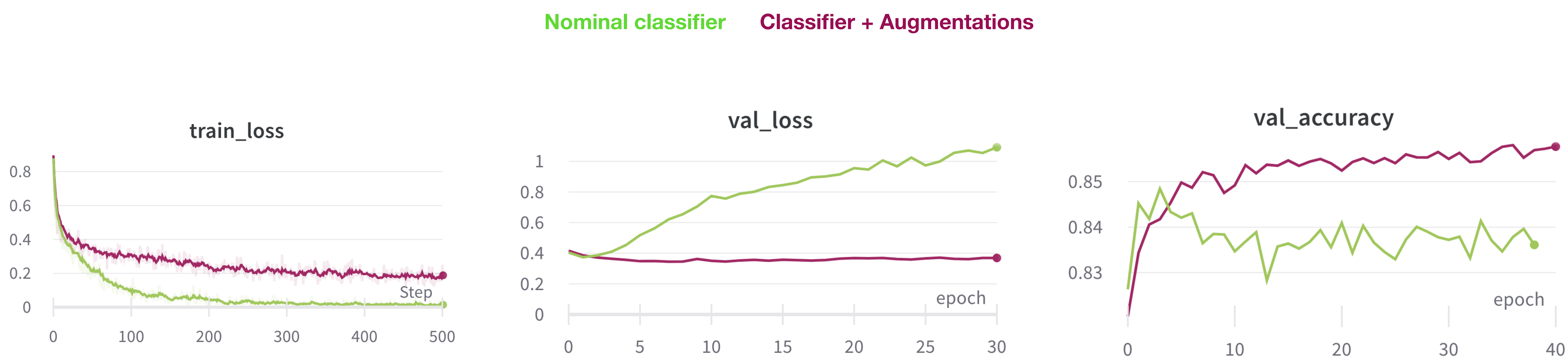
We only need to train the base model **once**!



- Can train **multiple** models cheaply
- All downstream models are **decorrelated** from the parameters we used for augmentations

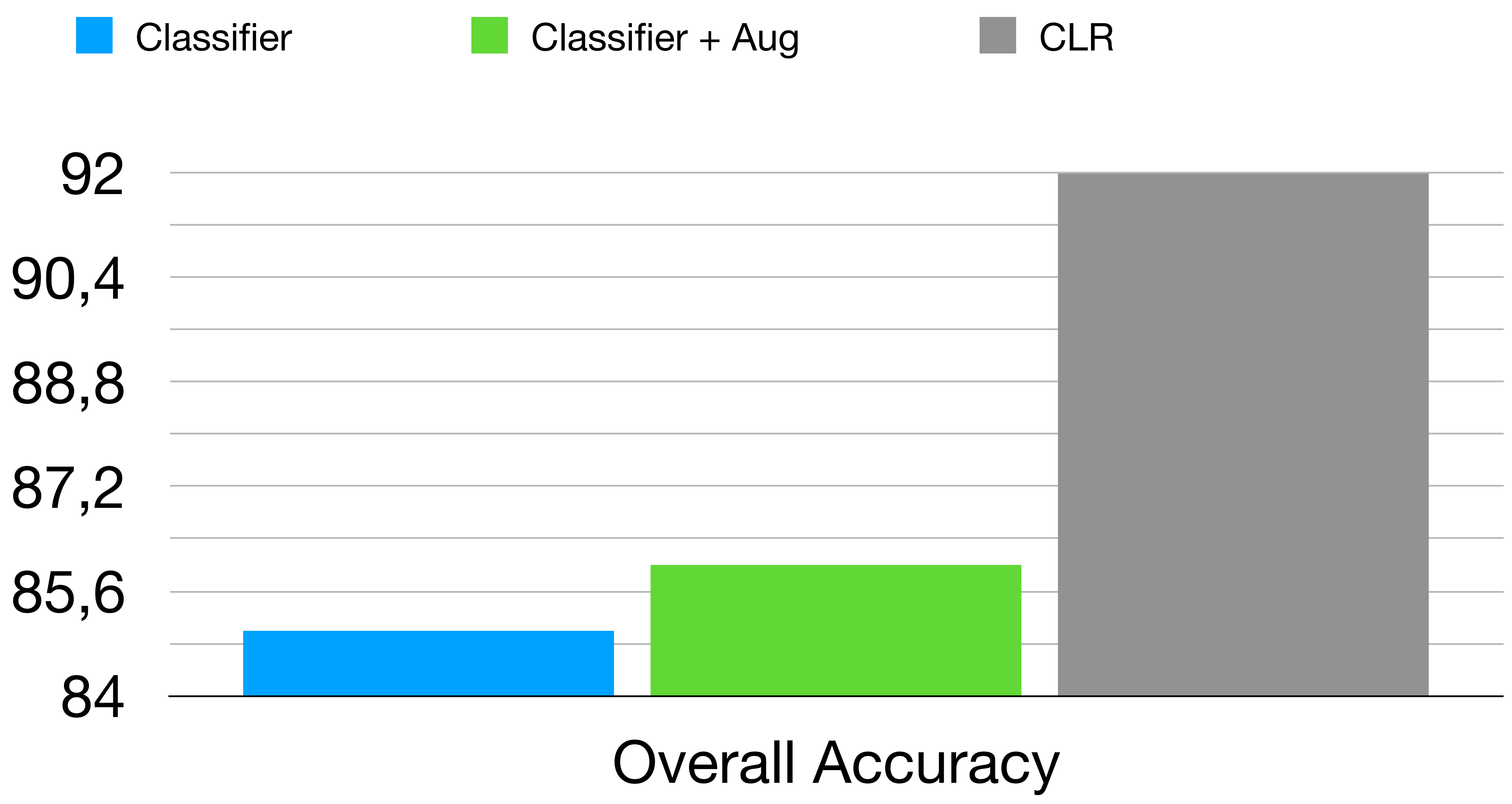
Preliminary Results

Classifier + Augmentations



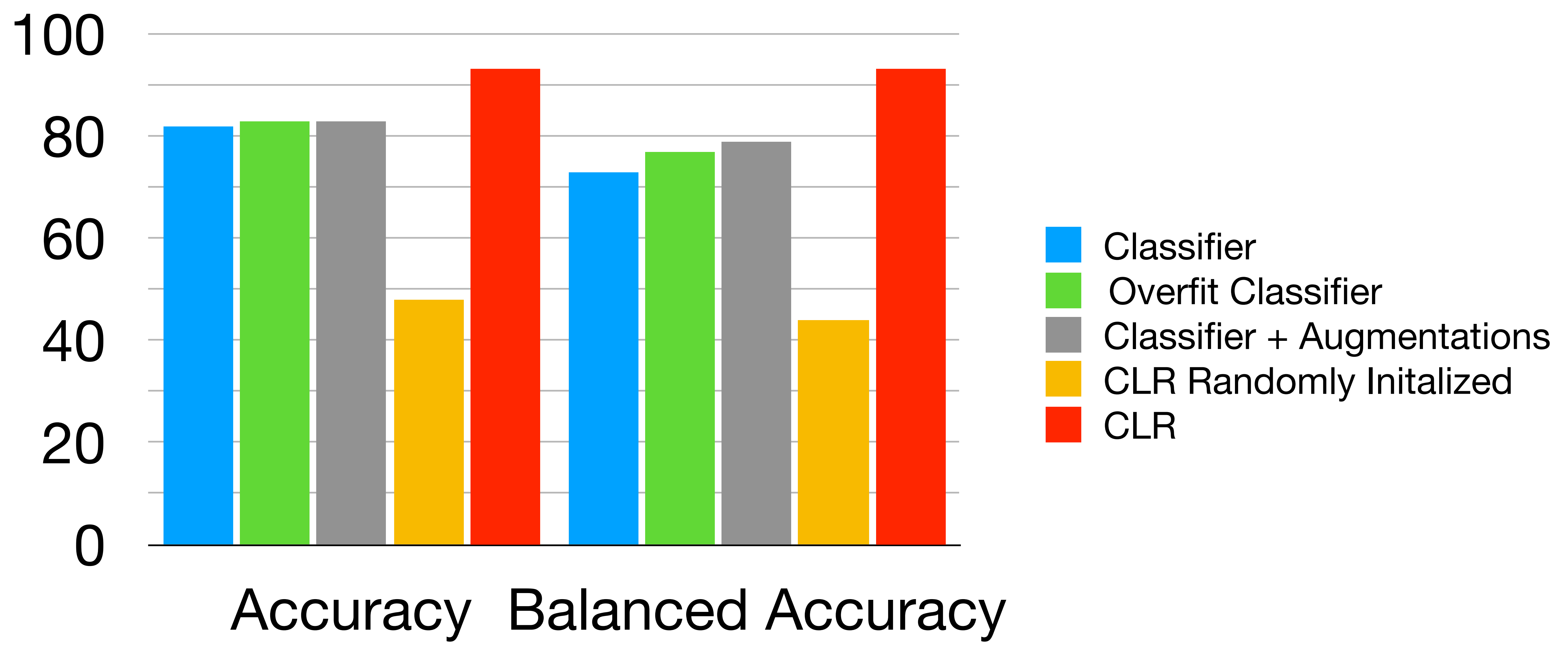
The augmentations **improve** our nominal classifier

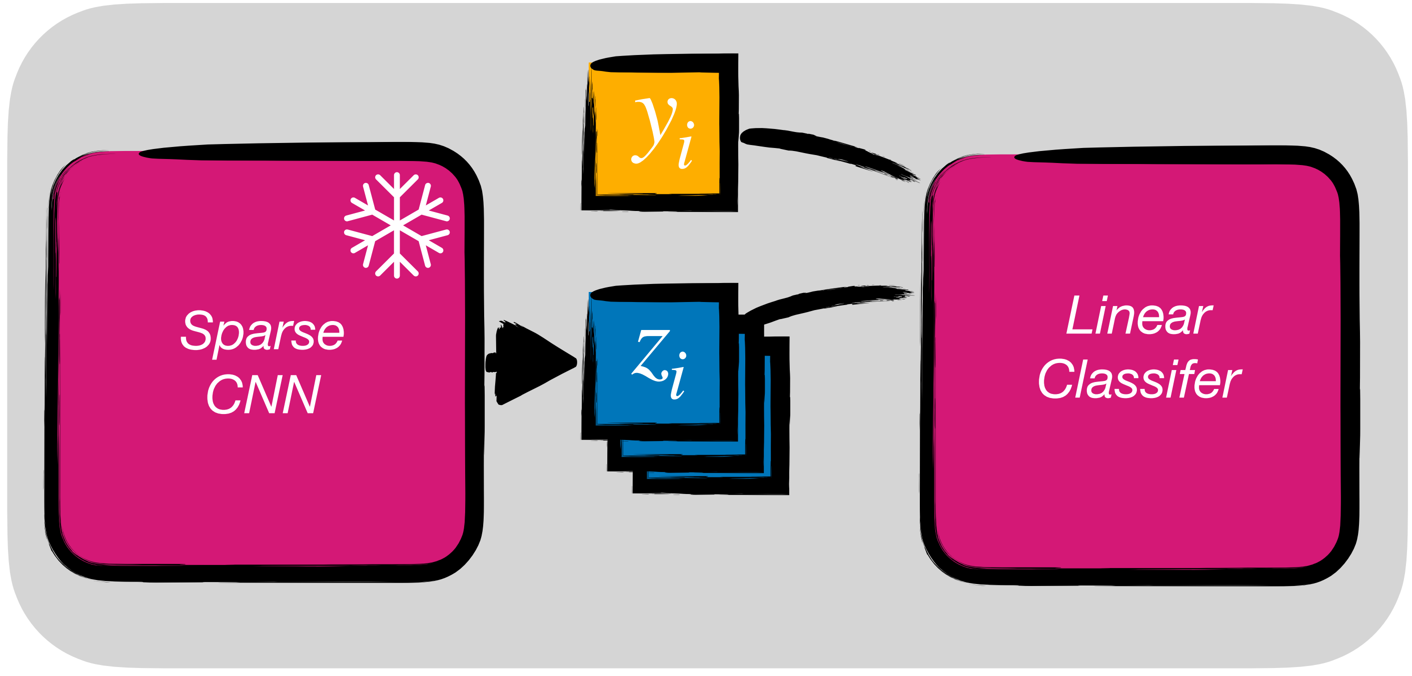
CLR Results



Classifier
Classifier + Aug
CLR
92
90,4
88,8
87,2
85,6
84
Overall Accuracy

CLR v Linear Classifer Baselines





Linear Classifier Test

All models are **frozen** - logistic regression fit on top.

For the classifiers the **last layer** is removed and we fit on the features after maxpooling.

For CLR we remove the **MLP** and again use the features after maxpooling.

More work needed

A lot more careful evaluation has to be done - however the results so far are very promising

Most importantly, how does this compare against:

- Other pretraining techniques - *MAE, data2vec*
- Other methods for decorrelation - *DANN, uncertainty aware learning*

Also would like to evaluate fine-tuning to other tasks e.g predicting the particles within an event

Future Work

- Consider a **more realistic scenario** - DUNE ND detector sim nuisance parameters as augmentations
- Use **larger batch** sizes for the base model
- Explore other contrastive learning methods

Future Work

- Fine-tune the model on another task e.g predicting **final state particles**
- Consider a **more realistic scenario** - DUNE ND detector sim nuisance parameters as augmentations
- Use **larger batch** sizes for the base model
- Explore other contrastive learning methods

I think this is could be a very exciting way to combine novel ideas from vision enhancing the way ML is used in physics analyses!

Thank you

radi.radev@cern.ch

Photo by Google DeepMind on Unsplash