Jet-Based Joint Embedding Predictive Architecture (J-JEPA)

CCNSB Seminar

Abhiram Tilak

CND Dual Degree
IIIT Hyderabad

January 22, 2025

Table of Contents

Introduction

About the Paper

This paper was showcased as a part of workshop ML4JETS2024 held on 4th November (Link). The paper was later published on 5th December 2024 to an Machine Learning conference called NeurIPS 2024.

Additional Information about the paper:

- This paper was an effort to adapt a well known Computer Vision Model called I-JEPA (Image-based Joint-Embedding Predictive Architecture) developed by Facebook Research in early 2023.

Introduction

About the Paper

This paper was showcased as a part of workshop ML4JETS2024 held on 4th November (Link). The paper was later published on 5th December 2024 to an Machine Learning conference called NeurIPS 2024.

Additional Information about the paper:

- This paper was an effort to adapt a well known Computer Vision Model called I-JEPA (Image-based Joint-Embedding Predictive Architecture) developed by Facebook Research in early 2023.
- This paper serves as concurrent work for the model our lab has been working on since last semester.

Background

Motivations

Problem with Supervised Models:

- Supervised learning typically acquire limited domain representations and focuses on a few key features for high prediction accuracy that must be learned anew for each task.

Self-Supervised Learning: A type of machine learning where models learn useful features and representations from unlabeled data
SSL aims to learn generic representations summarizing domain features that prove useful across various downstream tasks. SSL tasks can be formulated on unlabeled data.

Background

Motivations

Problem with Supervised Models:

- Supervised learning typically acquire limited domain representations and focuses on a few key features for high prediction accuracy that must be learned anew for each task.
- The problem comes when we have to scale the data being fed to the model.

Self-Supervised Learning: A type of machine learning where models learn useful features and representations from unlabeled data
SSL aims to learn generic representations summarizing domain features that prove useful across various downstream tasks. SSL tasks can be formulated on unlabeled data.

Background

Motivations

Problem with Supervised Models:

- Supervised learning typically acquire limited domain representations and focuses on a few key features for high prediction accuracy that must be learned anew for each task.
- The problem comes when we have to scale the data being fed to the model.
- A significant drawback of this is that the performance of ML models trained on simulations may not translate to real data, especially due to mismodeling in the former.

Self-Supervised Learning: A type of machine learning where models learn useful features and representations from unlabeled data
SSL aims to learn generic representations summarizing domain features that prove useful across various downstream tasks. SSL tasks can be formulated on unlabeled data.

Background

Motivations

Problem with Supervised Models:

- Supervised learning typically acquire limited domain representations and focuses on a few key features for high prediction accuracy that must be learned anew for each task.
- The problem comes when we have to scale the data being fed to the model.
- A significant drawback of this is that the performance of ML models trained on simulations may not translate to real data, especially due to mismodeling in the former.
- There have been efforts to run supervised models on huge datasets like ParT and OmniLearn-Alpha.

Self-Supervised Learning: A type of machine learning where models learn useful features and representations from unlabeled data
SSL aims to learn generic representations summarizing domain features that prove useful across various downstream tasks. SSL tasks can be formulated on unlabeled data.

Background

Motivations for JEPA

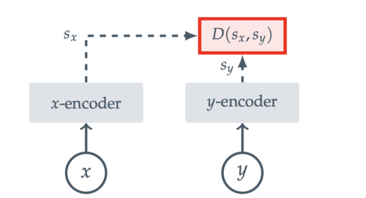
Contrastive Methods (Also called Joint Embedding Architecutre):

Contrastive Learning is a machine learning technique that teaches computers to understand similarities and differences by comparing pairs of data points.

It requires us to make several Data augmentations to the inputs, before we try to compare the difference between them.

These pretraining methods can produce representations of a high semantic level, but they also introduce strong biases that may be detrimental for certain downstream tasks or even for pretraining tasks with different data distributio

There have been a bunch of models in 2024 that use Contrastive methods like JetCLR, AnomalyCLR, BlackCLR, RS3L etc.



(a) **Joint-Embedding Architecture**

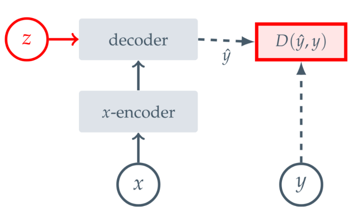
Background

Motivations for JEPA

Generative Methods: This idea is at the core of self-supervised generative methods, which remove or corrupt portions of the input and learn to predict the corrupted content. In particular, mask-denoising approaches learn representations by reconstructing randomly masked patches from an input, either at the pixel or token level.

Masked pretraining tasks require less prior knowledge than view-invariance approaches and easily generalize beyond the image modality.

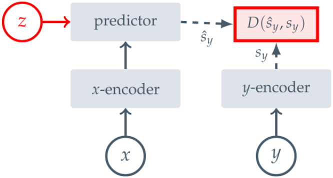
Only problem here is that they tend to underperform because they build only lower level semantic relationships between jet-representations.



(b) **Generative Architecture**

Architecutre

JEPA

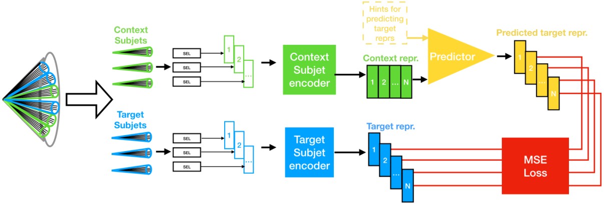


(c) **Joint-Embedding Predictive Architecture**

Given a jet, we recluster it into subjets, masking some as "target" subjets and defining others as "context" subjets. Then, we train a model to predict the representations of target subjets based on the representations of context subjets, using the positions of the target subjets as joint information.
This is also called Weakly-supervised model since the prediction stage uses some representations like positions of the target subjects and help the prediction.

Architecture

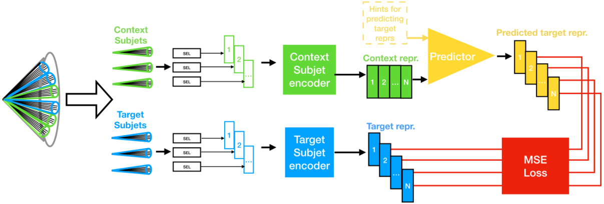
J-JEPA



- The J-JEPA architecture begins by splitting the large-radius jet (large black cone) into target subjets and context subjets.

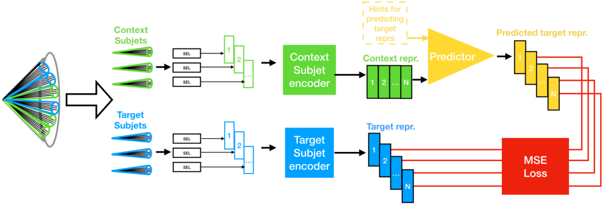
Architecture

J-JEPA



- The J-JEPA architecture begins by splitting the large-radius jet (large black cone) into target subjets and context subjets.
- The context encoder and target encoder then separately generate representations for the context subjets and the target subjets

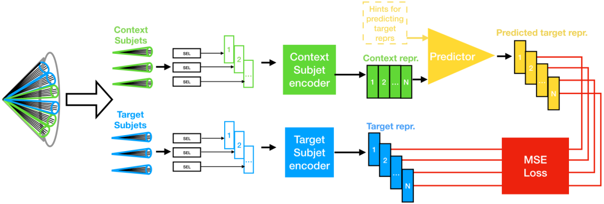
Architecture
J-JEPA



- The J-JEPA architecture begins by splitting the large-radius jet (large black cone) into target subjets and context subjets.
- The context encoder and target encoder then separately generate representations for the context subjets and the target subjets
- Using the positions of the target subjets as additional information (hints), the predictor takes the context representations and predicts the representations of the target subjets.

Architecture

J-JEPA



- The J-JEPA architecture begins by splitting the large-radius jet (large black cone) into target subjets and context subjets.
- The context encoder and target encoder then separately generate representations for the context subjets and the target subjets
- Using the positions of the target subjets as additional information (hints), the predictor takes the context representations and predicts the representations of the target subjets.
- Finally, the L2 loss function is used to compare the predicted target subjet representations with the encoded target subjet representations, minimizing the difference between them

Datasets and Training

Datasets:

Pretraining: JetClass: The pretraining task uses 1 Million jets, which is 1% of Jetclass dataset. It consists of 500k Top jets and 500k QCD jets.
Finetuning: TopTagging Reference Dataset: There are two scenarios in fine-tuning in this model. Both of them use TopTagging Dataset. The full dataset consists for 1.2 M Top jets.
Another scenario is using 10% of TopTagging dataset (120k jets). This is to represent situations where labeled training samplesare limited.

Training: Experiments were performed on a single NVIDIA A100 GPU. For

- AdamW optimizer with cosine decay

Datasets and Training

Datasets:

Pretraining: JetClass: The pretraining task uses 1 Million jets, which is 1% of Jetclass dataset. It consists of 500k Top jets and 500k QCD jets.
Finetuning: TopTagging Reference Dataset: There are two scenarios in fine-tuning in this model. Both of them use TopTagging Dataset. The full dataset consists for 1.2 M Top jets.
Another scenario is using 10% of TopTagging dataset (120k jets). This is to represent situations where labeled training samplesare limited.

Training: Experiments were performed on a single NVIDIA A100 GPU. For

- AdamW optimizer with cosine decay
- learning rate of 10^{-3}, and weight decay 10^{-2}

Datasets and Training

Datasets:

Pretraining: JetClass: The pretraining task uses 1 Million jets, which is 1% of Jetclass dataset. It consists of 500k Top jets and 500k QCD jets.
Finetuning: TopTagging Reference Dataset: There are two scenarios in fine-tuning in this model. Both of them use TopTagging Dataset. The full dataset consists for 1.2 M Top jets.
Another scenario is using 10% of TopTagging dataset (120k jets). This is to represent situations where labeled training samplesare limited.

Training: Experiments were performed on a single NVIDIA A100 GPU. For

- AdamW optimizer with cosine decay
- learning rate of 10^{-3}, and weight decay 10^{-2}
- Epochs: 80, batch-size: 64

Evaluation

Metrics:

- Compare finetuned model, with model trained from scratch (no pretraining).

Other Evaluations:

Evaluation

Metrics:

- Compare finetuned model, with model trained from scratch (no pretraining).
- Prediction Accuracy (number of correct classifications vs wrong)

Other Evaluations:

Evaluation

Metrics:

- Compare finetuned model, with model trained from scratch (no pretraining).
- Prediction Accuracy (number of correct classifications vs wrong)
- Background rejection at signal efficiency 50%.

Other Evaluations:

Evaluation

Metrics:

- Compare finetuned model, with model trained from scratch (no pretraining).
- Prediction Accuracy (number of correct classifications vs wrong)
- Background rejection at signal efficiency 50%.

Other Evaluations:

- Using Traditional Embeddings vs. Custom Attension based embeddings

Evaluation

Metrics:

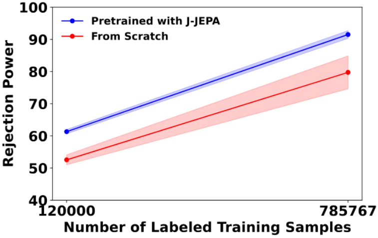
- Compare finetuned model, with model trained from scratch (no pretraining).
- Prediction Accuracy (number of correct classifications vs wrong)
- Background rejection at signal efficiency 50%.

Other Evaluations:

- Using Traditional Embeddings vs. Custom Attension based embeddings
- Flattened subjets vs. Using class attention blocks to aggregate subject representations.

Results

Model	Aggregation	Baseline 10%	Baseline Full	Finetuned 10%	Finetuned Full
		Accuracy [%]			
SjT-T	Flatten	87.52 ± 0.16	89.13 ± 0.10	88.21 ± 0.55	89.95 ± 0.13
SjT-T	Cls Attn	88.30 ± 0.18	89.67 ± 0.13	88.67 ± 0.02	90.00 ± 0.07
AE-SjT-T	Flatten	88.92 ± 0.15	90.01 ± 0.08	$\mathbf{88.94 \pm 0.13}$	$\mathbf{90.03 \pm 0.07}$
AE-SjT-T	Cls Attn	88.84 ± 0.21	$\mathbf{90.03 \pm 0.05}$	88.82 ± 0.11	90.00 ± 0.12
		$1/\varepsilon_B(\varepsilon_S = 0.5)$			
SjT-T	Flatten	40.50 ± 1.26	70.70 ± 1.46	53.67 ± 9.97	90.06 ± 3.80
SjT-T	Cls Attn	52.56 ± 1.54	79.75 ± 5.12	61.32 ± 0.66	91.51 ± 1.20
AE-SjT-T	Flatten	67.34 ± 1.40	97.79 ± 3.90	$\mathbf{70.47 \pm 1.09}$	97.52 ± 1.71
AE-SjT-T	Cls Attn	67.19 ± 1.54	$\mathbf{99.38 \pm 2.80}$	68.25 ± 1.64	95.47 ± 1.83



Conclusion

This paper will likely have a follow-up which scales the data up and uses different encoders and uses various different datasets.
The future work involves:

- Implementing physics-informed architectures for the context and target encoders, such as the Particle Transformer

Conclusion

This paper will likely have a follow-up which scales the data up and uses different encoders and uses various different datasets.
The future work involves:

- Implementing physics-informed architectures for the context and target encoders, such as the Particle Transformer
- Alternative strategies for embedding and defining targets and context

Conclusion

This paper will likely have a follow-up which scales the data up and uses different encoders and uses various different datasets.
The future work involves:

- Implementing physics-informed architectures for the context and target encoders, such as the Particle Transformer
- Alternative strategies for embedding and defining targets and context
- Generalize the JEPA scheme to different physics objects: particles, events, detector readout, etc.

Thank You