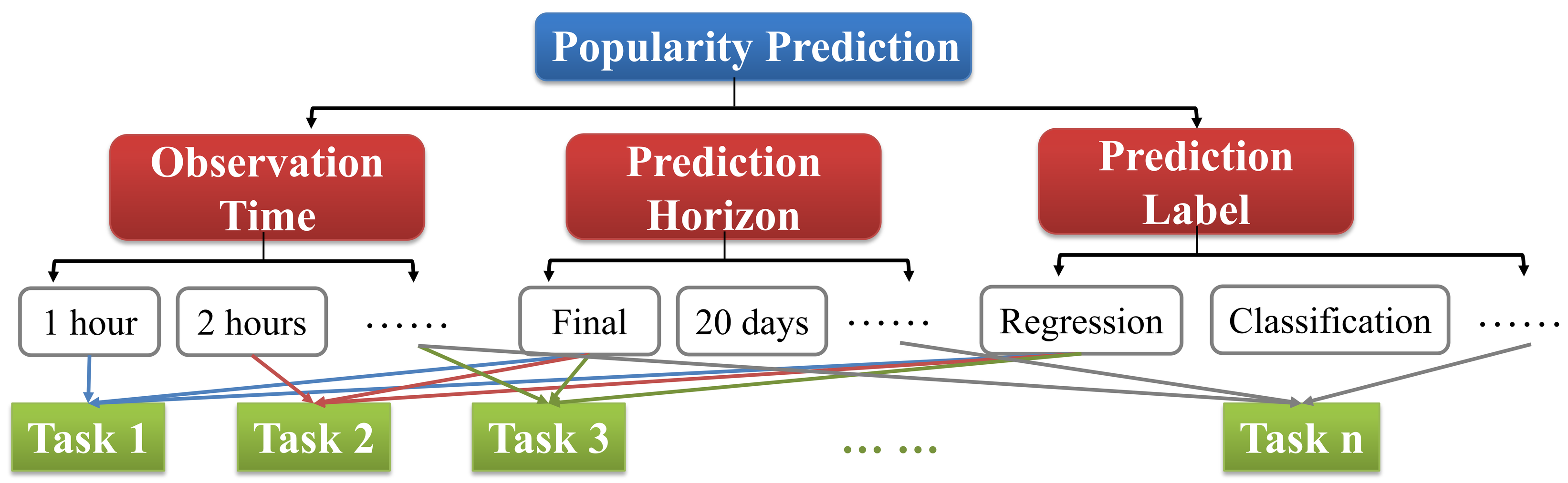
PREP: Pre-training with Temporal Elapse Inference for Popularity Prediction

Qi Cao, Huawei Shen, Yuanhao Liu, Jinhua Gao, Xueqi Cheng
{caoqi, shenhuawei, liuyuanhao20z, gaojinhua, cxq}@ict.ac.cn
Data Intelligence System Research Center, Institute of Computing Technology, Chinese Academy of Sciences

1. Motivation

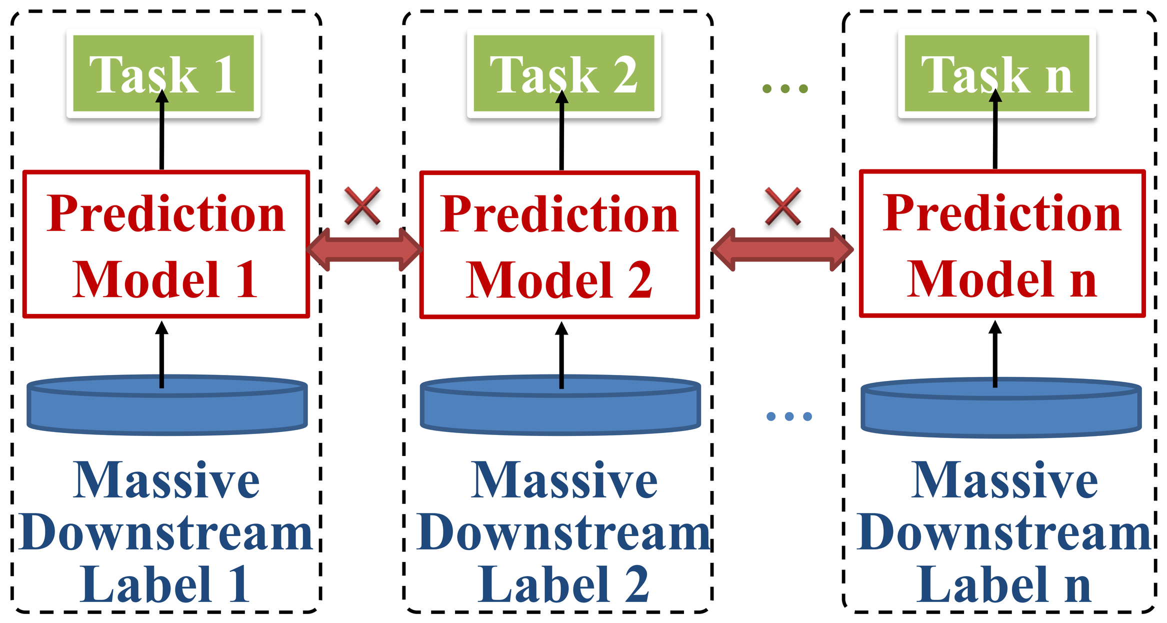
1.1 Various Popularity Prediction Settings

- **There are various popularity prediction tasks settings in different situation**, e.g., varying length of observation time or prediction horizon, different types of prediction label...



1.2 Existing Paradigm: Separate Training

- Existing methods generally train a separate prediction model for each prediction task



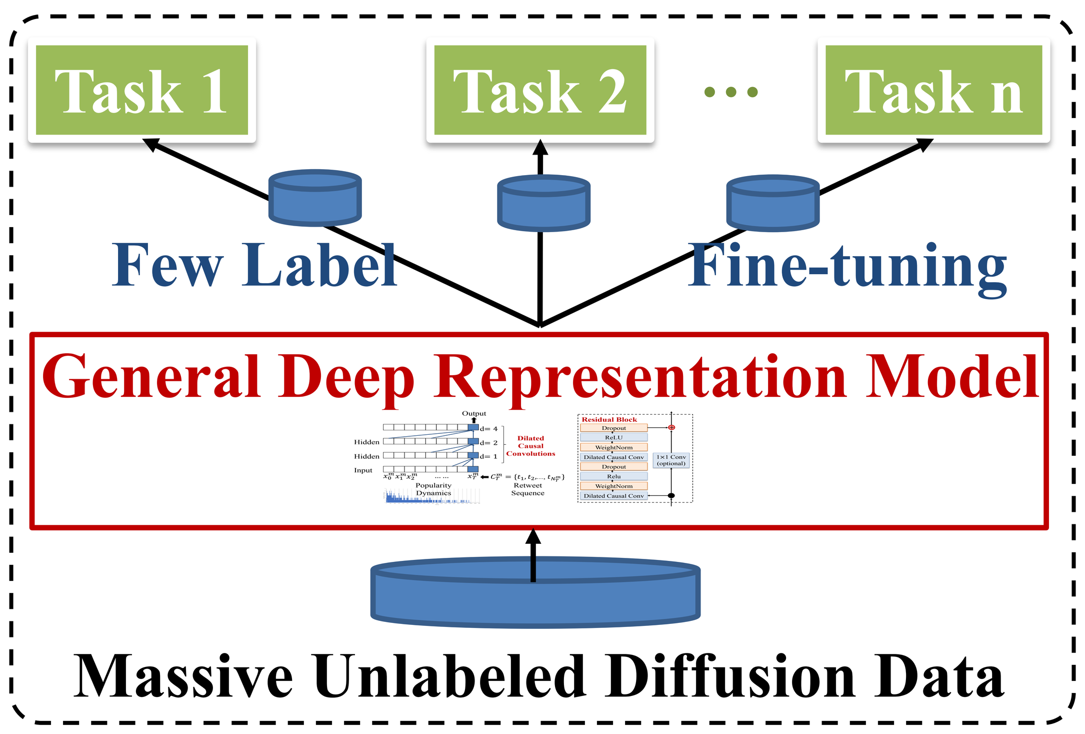
- **Drawbacks**: Causing a great waste of training time and computational resources.

There still lacks a both effective and efficient popularity prediction model that can handle various task settings

2. Method

2.1 Pre-training Framework

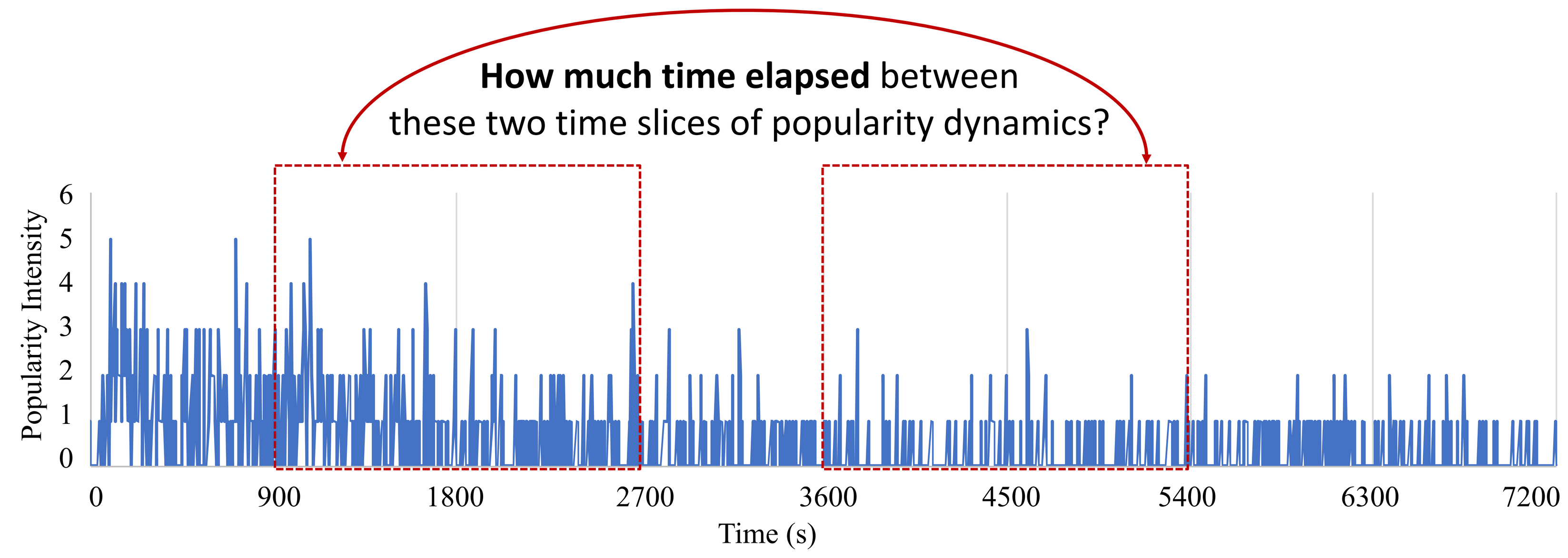
- PREP: Pre-training a general deep representation model



- **Advantages:** The pre-trained model can be **effectively and efficiently** transferred into various popularity prediction tasks

2.2 Pretext Tasks for Pre-training: Temporal Elapse Inference

- Randomly samples pairs of time slices of popularity dynamics and aims to infer the time elapsed between these two time slices



- **Intuition Behind:** the pre-trained model have to understand the temporal context information and capture the evolution pattern of popularity dynamics, which is critical for downstream popularity prediction tasks

3. Experiment

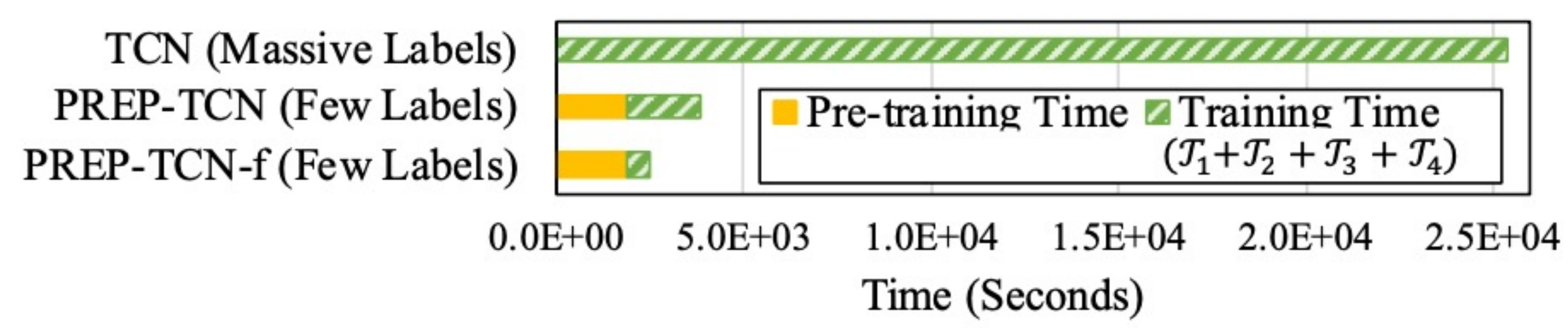
3.1 Prediction Performance

	Task $\mathcal{T}_1$		Task $\mathcal{T}_2$		Task $\mathcal{T}_3$		Task $\mathcal{T}_4$	
Methods	MRSE	R-Acc	MRSE	R-Acc	MRSE	R-Acc	C-Acc	F1
Separate Training on Weibo with Massive Labels								
Seismic	-	-	-	35.1%	-	37.5%	52.4%	0.508
DeepHawkes	0.510	35.7%	0.379	38.9%	0.342	40.7%	49.8%	0.000
CasCN	0.347	40.8%	0.372	38.4%	0.326	44.5%	65.4%	0.664
Feature-based	0.251	42.6%	0.212	43.9%	0.172	53.6%	59.2%	0.690
TCN	**0.232**	**47.4%**	**0.175**	**52.4%**	**0.137**	**63.1%**	**73.6%**	**0.713**
Transfer Pre-trained (or random initialized) model on Weibo with Few Labels								
TCN-f	0.809	0.4%	0.396	25.2%	0.396	25.2%	49.7%	0.000
PREP-TCN-f	0.322	33.5%	0.258	40.1%	0.236	44.1%	66.6%	0.645
TCN	0.262	43.8%	0.191	51.0%	0.168	57.0%	68.1%	**0.674**
PREP-TCN	**0.238**	**47.8%**	**0.184**	**51.7%**	**0.147**	**61.4%**	**70.9%**	0.669
Separate Training on Twitter with Massive Labels								
Seismic	-	-	-	60.7%	-	66.4%	61.4%	0.520
Feature-based	0.077	77.8%	0.106	70.6%	0.084	77.9%	65.3%	0.582
TCN	**0.054**	**82.3%**	**0.086**	**74.3%**	**0.063**	**81.9%**	**70.9%**	**0.634**
Transfer Pre-trained (or random initialized) model on Twitter with Few Labels								
TCN-f	0.238	40.7%	0.258	37.7%	0.258	37.7%	54.8%	0.000
PREP-TCN-f	0.166	53.1%	0.192	48.0%	0.217	46.1%	65.6%	0.534
TCN	0.073	76.1%	0.100	70.6%	0.084	76.7%	**70.7%**	0.614
PREP-TCN	**0.057**	**83.0%**	**0.090**	**71.9%**	**0.069**	**79.9%**	70.6%	**0.630**

- When transfer the pre-trained model into downstream tasks with few labels, our pre-trained model (PREP-TCN) significantly outperform the random initialized TCN model
- The PREP-TCN which is fine-tuned with few downstream labels even achieves comparable prediction performance when compared with TCN trained with massive downstream labels under the paradigm of separate training

3.2 Efficiency

- Even taking into account the time of pre-training, the pre-training framework (PREP-TCN) is much more efficient than separately trained TCN



The code is publicly available in Github (https://github.com/CaoQi92/PREP). More details please refer to our paper.