**NexTSS**

**User Manual**

Ashita Jawali, Parker Simpson, Sanat Mishra, Sofia Lima

# **About this Pipeline**

The pipeline presented here uses a deep learning-based approach to predict TSS (Transcription Start Sites) from raw genomic data in *Mus musculus*. The model architecture is inspired by DeepTSS - a state of the art convolutional network that predicts transcription start sites. The model in our project was trained on data from *Homo sapiens* TSS. It operates on genomic windows 600 bases in length. The input consists of the encoded raw genomic sequences, DNA structural features, and Genomic Signal Processing (GSP) features. The training data consists of an equal number of positively and negatively labeled data for the presence or absence of TSS. CAGE-seq data is considered the putative positive label for TSS. The pipeline outputs a probabilistic outcome for each test sample depending on whether a TSS is found or not.

# **Significance of problem**

Our deep learning model is designed to predict transcription start sites (TSSs) in genomic DNA sequences by increasing signal-to-noise ratio from CAGE-seq data. TSS prediction is of paramount importance in the field of genomics and molecular biology, as it provides insights into the regulation of gene expression and the organization of the genome, leading to a deeper understanding of cellular processes and the development of novel therapeutic strategies. The sequences contain information that can accurately predict a transcription start site. By using a deep learning approach, the model architecture is able to capture intricacies of sequence features that are highly informative while using the CAGE-seq data to identify putative TSS regions.

# **Workflow**

The workflow broadly consists of the three phases - i) Preprocessing, ii) Feature Extraction, and iii) Modeling as seen in Fig 1. The pipeline can be executed with a few commands provided in “Running the pipeline” section with all required data and scripts available on github: <https://github.com/psimps21/03713_tss_prediction>

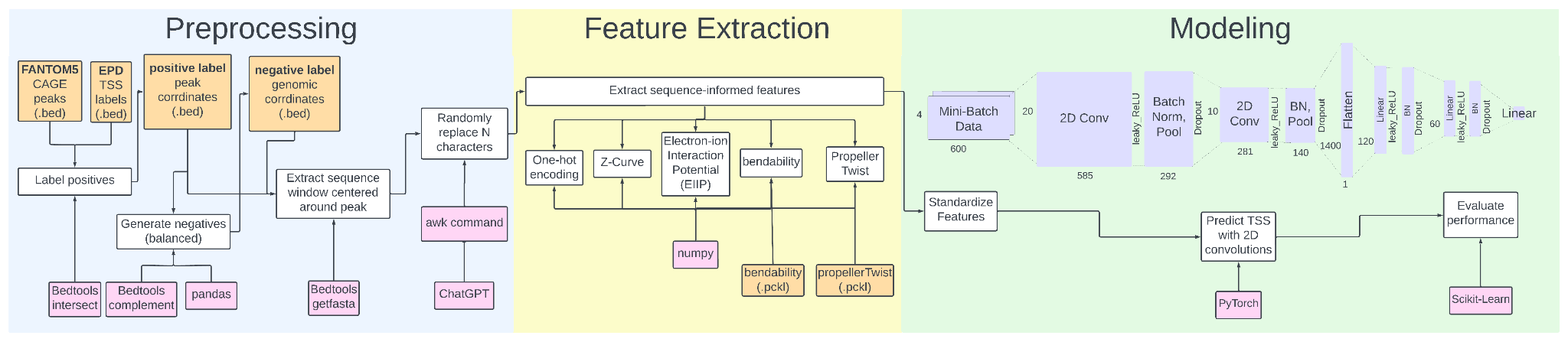


Fig. 1: The complete workflow from preprocessing the data to making TSS predictions. White boxes are the computational objectives. Orange boxes are the datafiles needed for the pipeline. Pink boxes are the tools used to accomplish the corresponding task. Each of the three steps is described in more detail in the subsequent paragraphs.

# **Required Packages / Tools**

The Python packages needed to run our pipeline can be found in the yaml file. These packages in their exact versions can be downloaded when setting up the Conda environment.

## **BEDTools**

Documentation and download instructions: <https://bedtools.readthedocs.io/en/latest/>

Bedtools is a toolkit for genome arithmetic and analysis of genomic features. It provides a comprehensive suite of utilities for comparing, intersecting, and manipulating genomic intervals in various file formats, such as BED, GFF/GTF, and VCF. Bedtools is widely used in genomic research for various tasks, including annotation, coverage calculation, and feature overlap analysis

## **Pytorch**

Documentation and download instructions: [PyTorch documentation — PyTorch 2.0 documentation](https://pytorch.org/docs/stable/index.html)

Pytorch is a deep learning toolkit to build and train neural networks. Using a GPU device requires cuda which is already installed on Bridges-2.

# **Running the pipeline**

This section contains the commands and files needed to execute the pipeline assuming the user has access to Bridges-2 at the Pittsburgh Supercomputing Center (PSC). It is recommended to download the github repository in the $PROJECT directory.

The sequence of commands are listed in Fig. 5 below.

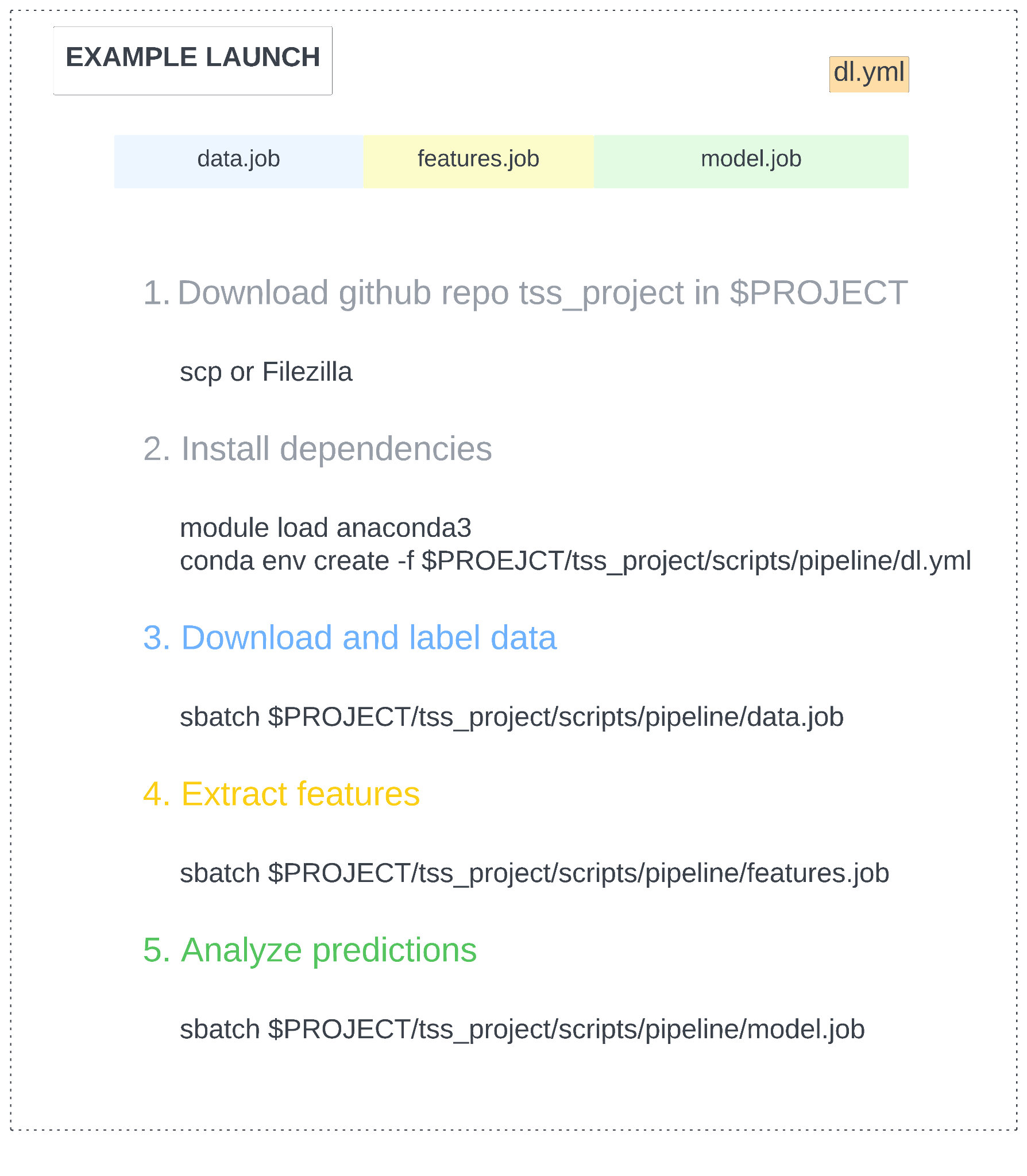


Fig. 5: The sequence of commands needed to execute the pipeline end-to-end.

# **Package Installation**

All packages needed for the pipeline can be installed with the following one-line command:

conda env create -f $PROJECT/tss\_project/scripts/pipeline/dl.yml

This .yml file contains the pytorch, pandas, and Scikit-learn dependencies required to run the pipeline.

## **Required input files:**

CAGE peak data files from FANTOM can be obtained with the ‘wget’ command, as written in the data.job script. EPD files can not be obtained this way and must exist in the working directory. This is accomplished by downloading the github repository.

## **Command Line**

The data preprocessing, feature extraction, and modeling can be accomplished by executing the three job scripts from the command line.

## **Example Output**

The output will consist of the training and testing evaluation plots, as well as a models directory that contains the trained model.