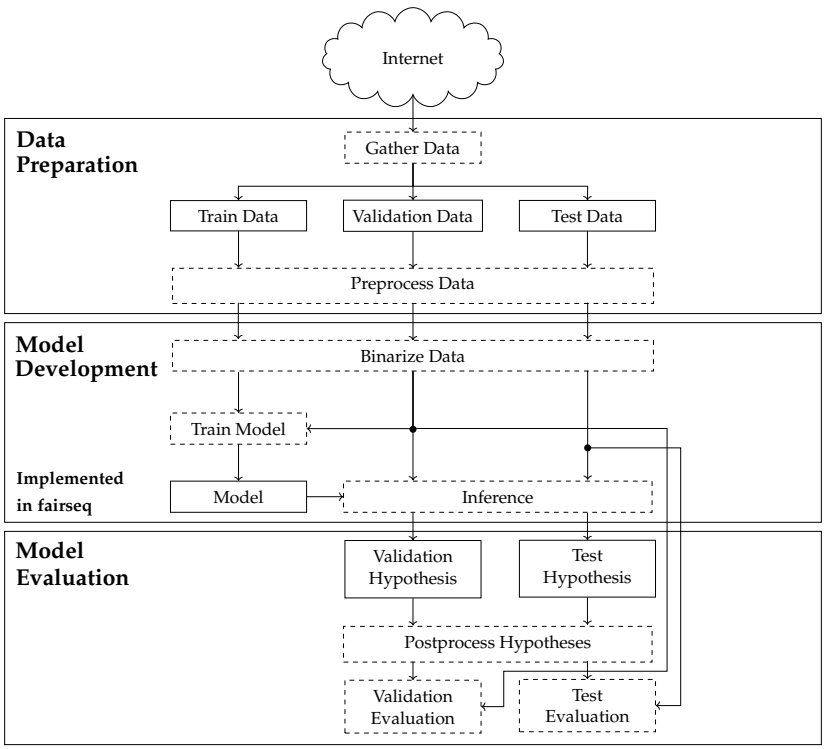

Figure 6.1: The general experimentation pipeline. Dashed rectangles denote processes, whereas plain bordered ones are objects obtained from said processes.

of the most popular open source ML frameworks. Similarly to its alternatives, it provides GPU accelerated tensor computing operations, as well as a set of implemented modules and an interface for easy definition of deep learning models through the usage of computational graphs for automatic differentiation.

Additionally, it incorporates its own training loops, predefined model architectures, functionalities for easy processing of sequence data for the training of big models in many GPUs, automatic saving of models at different checkpoints when training a model, as well as a robust logging system that enables researchers to quickly learn about potential problems in the training process.

Apart from the python API, fairseq offers command line tools that facilitate the development of large models, as well as the possibility to extend the toolkit by defining new models, metrics, etc. The main command line tools that will be used throughout this work are:

- `fairseq-preprocess`. It takes (preprocessed) parallel text as input and builds the vocabularies in a format that the model can parse, binarizing the training data for ease of use by the model. It takes into consideration a set of command line arguments and supports parallel computing for large datasets.
- `fairseq-train`. It trains the model on one or many GPUs. providing a way to configure the model and set the hyperparameters to use for training, as well as the binarized data path from `fairseq-preprocess`. It offers an implementation of the Transformer model as described in Section 2.3.3
- `fairseq-interactive`. It implements the beam search algorithm presented in Section 2.3.4, translating data with a trained model in inference mode. Within its parameters, we can specify the number of beams, inference batch size, the model to use, etc.