# Baseline: Strong, Extensible, Reproducible Deep Learning Baselines for NLP

Daniel Pressel, Brian Lester, Sagnik Ray Choudhury, Matt Barta, Yanjie Zhao, Amy Hemmeter

## Baseline: Rapid Development for Deep NLP Research

- Tracks experiments automatically, makes research reproducible
  - Models, hyper-parameters, metrics, datasets, embeddings
- Rapid development and experiment cycle, supports operational deployment
  - Efficient, extensible training
  - Reusable components and models to build on
- Multiple Deep Learning framework support

## Baseline: Development Philosophy

- Keep it Simple
  - Minimal dependencies, effective design patterns
  - A la carte design: take only what you need
- Stronger baselines lead to stronger models
- Research should be reproducible
- Building your own training and evaluation pipeline is time consuming and error-prone, provide this for users
- Make it easy to deploy models operationally

## Baseline: Tasks and Models

- Strong, deep baselines for common NLP tasks
    - Classification, Tagging, Encoder-Decoders, Language Modeling
- We support your favorite DL framework
    - TensorFlow, PyTorch and DyNet
- Reusable components to build your own SoTA models
    - Addons with SoTA models, contextual embeddings

## Baseline: Deep Learning Pipeline

- Built-in dataset and embedding downloads
- Configuration-based, extensible training
- Metrics reporting
    - Metrics defined for each task
    - TensorBoard, Visdom support
- XPCTL (Experimental Control): A leaderboard to track progress and configurations
- HPCTL (Hyper-Parameter Control): A hyper-parameter tuner for finding your best model

## Baseline Design Patterns

- Embeddings as model sub-graphs
  - Embeddings are self-contained objects
  - Model design orthogonal to Embedding, extend by configuration
- Embeddings specified in config file
- Models provide basic idioms for overriding sub-components
  - Encoders/Pooling, Decoders, Embeddings, Stacking
- Framework-level Blocks to easily build new models from scratch

## Experiment Control (XPCTL)

- Tracks experimental results from model runs in store
  - Stores original configuration, all metric events
  - Store can be local or centralized
  - Pluggable backends for data storage
- Quickly aggregate statistics on multiple runs
- Query leaderboard to see your model’s performance on a tasks

## Hyperparameter Control (HPCTL)

- Tune hyperparameters by specifying a search template.
- Supports many search techniques.
    - Random, Grid, Uniform, Normal
- Coordinates the running of models in parallel
    - Allocates GPUs and prevents collisions
    - Supports running models inside a docker container or as a python multiprocess
- Aggregates results and can automatically store runs in XPCTL

## Want to help build?

- PRs welcome!
- Codebase:
    - https://github.com/dpressel/baseline
- Public addons:
    - https://github.com/dpressel/baseline/tree/master/python/addons
- Contact Info
    - dpressel@gmail.com, @DanielPressel