Documentation for Wide and Deep Bandits

Implementations:

- **wide_deep_bandits.py** – A simple implementaton of the wide and deep model. Actions selected greedily. Can be used with GPU.
    - **Model combination methods** - The wide network trains embedding using the user IDs, and the deep network consist of multiple linear layers. The wide and deep components can be combined using several different methods:
        - **Add Rewards**: The predicted rewards from the wide and deep network are added in the last layer and loss is computed on the sum.
        - **Concatenate Rewards**: The predicted rewards from the wide and deep network are concatenated and loss is computed on the concatenation.
        - **Linear Combination of Rewards**: The predicted rewards from the wide and deep network are combined using a final linear layer.
        - **Last Layer Representation**: The embedding from the wide model is concatenated with the last layer representation of the deep neural network. A final linear layer then uses the concatenated representations to predict the reward.

- **wide_deep_bandits_BLR_TS.py** – This version uses Bayesian Linear Regression and Thompson Sampling similar to the original space-bandits. Actions can be selected using one of the following methods:
    - **BLR**: Use the expected rewards from the Bayesian linear regression to predict best action.
    - **BLR+TS**: Rewards are sampled from the posterior distribution using Thompson Sampling.
    - **Forward**: Use the predicted rewards from the neural networks to select the actions directly.

Parameters:

- **num_actions** – Number of actions (int). Required.
- **num_features** – Numbers of features in context (int). Required.
- **wide_embed_size** – Size of embedding dictionary for the wide model (int, default 100)
- **wide_embed_dim** – Dimension of embedding for the wide model (int, default 64)
- **wd_combine_method** – Method for combining the wide and deep models in the wide+deep model (string, possible values listed below):
    - "add_rewards" - the predicted rewards from the wide and deep network are added in the last layer and loss is computed on the sum.
    - "concat_reward" - the predicted rewards from the wide and deep network are concatenated and loss is computed on the concatenation.
    - "concat_reward_llr" - the predicted rewards from the wide and deep network are combined using a final linear layer.
    - "concat_representation_llr" - the embedding from the wide model is concatenated with the last layer representation of the deep neural network, a final linear layer then uses the concatenated representations to predict the reward (default).
- **update_freq_nn** – Frequency to update the model, default updates model for every data point (int, default 100)
- **do_scaling** – Whether to scale the contexts (bool, default True)
- **num_epochs** – Number of steps to Train for each update (int, default 1)
- **max_grad_norm** – maximum gradient value for gradient clipping (float, default 5.0)
- **initial_lr_wide** – initial learning rate for wide network training (float, default 0.01)
- **initial_lr_deep** – initial learning rate for deep network training (float, default 0.01)
- **lr_decay_rate_wide** – learning rate decay for wide network updates (float, default 0.0)
- **lr_decay_rate_deep** – learning rate decay for deep network updates (float, default 0.0)
- **reset_lr** – whether to reset learning rate when retraining network (bool, default True)
- **batch_size** – size of mini-batch to train at each step (int, default 512)

Parameters only in wide_deep_bandits_BLR_TS.py:

- **a0** – initial alpha value (int, default 6)
- **b0** – initial beta_0 value (int, default 6)
- **lambda_prior** – lambda prior parameter (float, default 0.25)