Datasheet For MetaCC Dataset

5.1 Motivation

Meta-learning offers data-efficient learning of novel tasks by following specifically designed training-frameworks on a distribution over tasks. Despite the steadily improved performance on standard image recognition tasks, there are several outstanding challenges to be addressed [13], including *generalisation to a wide range of tasks* and *when there is distribution shift between train and test*. Systematic study of these issues is hampered because conventional benchmarks do not provide a way to quantitatively measure or control the complexity or similarity of task distributions. MetaCC and the corresponding dataset contribute to the future study of these issues by introducing a channel coding meta-learning benchmark, which enables finer control and measurement of task-distribution complexity and shift for study.

Our MetaCC benchmark provides a number of benefits to the community: (i) It provides a systematic framework to evaluate future meta-learner performance with regards to under-fitting complex task distributions and robustness to train-test task distribution shift that is ubiquitous in real use cases such as sim-to-real. Both of these are crucial challenges which must be addressed for meta-learners to be of practical value in real applications. (ii) MetaCC has the further advantage of being independently elastic in every dimension. Future studies can thus use it to study impact of number of tasks, instances or categories; dimension of inputs; difficulty of tasks, width of task distributions and train-test distribution shift. Unlike existing saturated small toy benchmarks, or large unwieldy benchmarks, these properties make it suitable for the full spectrum of research from fast prototyping to investigating the peak scalability of meta-learners. (iii) By addressing rapid adaptation to new domains, MetaCC complements existing multi-task focused meta-learning benchmarks. This means that meta-learners are challenged to beat strong baselines including ERM and classic Viterbi algorithms. With regard to robustness, this will allow meta-learners to ultimately be compared directly against methods that improve the ERM baseline through improving robustness to domain-shift. (iv) Finally, MetaCC directly instantiates a task of significant real-world importance, where advances will immediately impact future communications systems

5.2 Composition

The channel coding dataset is generated using script provided in the benchmark code repository. A range of attributes, including but not limited to channel parameters, message length, code rate, encoding algorithm, can be customized shall users wish to produce their own version of the dataset. The dataset consists of ground truth messages (labels) and noisy code bits as an input to the decoder (inputs), under a given channel conditions. The labels are binary digits of block length 10, and we use convolutional encoder with code rate 1/2, which leads to inputs of continuous values in shape of 10x2 for each instance. The pre-generated dataset consists of a train and a test set, and the train set consists of 21 sub-set, each saved in an .npy file, corresponding to a single or multiple channel families (e.g. AWGN, Bursty) within a predefined range of channel parameters (e.g. wide, narrow) specified in the corresponding json file. For each sub-set, there are 100 noise settings (e.g. AWGN SNR=0) sampled uniformly from the defined range, each setting contains 1000 uniformly sampled ground truth messages from all possible combinations (2^{10}), and 20 noisy input instances produced based on the message and channel parameters. The test set consists of 10 noise settings of fixed step length within the 'Expanded' range in each noise family, 100 ground truth messages, and 50 corrupted noisy instances of each message. Since the nature of channel coding data are randomly synthesized binary digits and based on well-established theoretical channel models, the dataset is self-contained and does not contain any confidential data.

5.3 Collection Process

The collection process of the synthetic dataset took around 3 hours on a single Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz. Both channel parameters and the ground truth labels are uniformly sampled and the random seed is set to 99. The real channel data collection used a wireless testbed setup consisting of two separate N200 USRPs as the transmitter and the receiver using antennas to communicate over air. The USRPs are connected to the system through the ethernet medium. We use MATLAB 2021 to preprocess and post-process the data while we use GNURadio to communicate with the USRPs. We derive the frame structure and the modulation parameters from the WiFi standard

802.11a. The transmit signals are arranged in frames with a preamble followed by data. The preamble consists of a short training sequence (STS) and a long training sequence (LTS). The encoded bits are mapped into 64-QAM symbols on the transmitter side and then modulated onto the subcarriers of the OFDM symbol along with the guard interval. The symbols also carry the pilot carriers in specific locations to aid in channel estimation and phase offset correction. These symbols are then converted to the time domain, appended by a cyclic prefix, and sent to the USRPs for transmission. Upon receiving the signal at the other USRP, we use the STS, and the LTS preambles for synchronization and frequency offset corrections. We remove the added cyclic prefix and convert the signal into frequency domain through an FFT. Lastly, we use the pilot carriers for channel equalization followed by demodulation to get the corresponding LLRs. The SNR of the transmission is managed by changing the transmit and receive power gains in order to achieve a requisite error performance mandated by the training procedure.

5.4 Preprocessing/cleaning/labeling

The data synthesis produce clean data with label, therefore no preprocessing/cleaning/labeling is needed.

5.5 Uses

Channel coding has been an established topic in communication theories. There are numerous channel coding studies, [16, 17] to name a few. However, this is the first proposed dataset for the purpose of benchmarking meta-learning algorithms. The dataset can also be used for future study of other machine learning algorithms.

5.6 Distribution

The dataset used in this benchmark study, the code used to generate the dataset, the benchmark implementation, and the documentation of the project is available on GitHub: https://github.com/ruihuili/MetaCC.git.

5.7 Maintenance

Primary supporting persons of the dataset and the benchmark code base:

- Rui Li (rui.li@samsung.com)
- Ondrej Bohdal (Ondrej.Bohdal@ed.ac.uk)
- Da Li (da.li1@samsung)

Main contact for testbed data related matters:

- Rajesh Mishra (rajeshkmishra@utexas.edu)

The benchmark repository is open to pull requests if others wish to contribute new algorithms and features.