Specification of Continual Few-Shot Learning Tasks – Version 1.0

Antreas Antoniou[1] Massimiliano Patacchiola[1] Mateusz Ochal[1] Amos Storkey[1]

1. Task Definitions

In continual few-shot learning (CFSL), a task consists of a sequence of (training) support sets $\mathcal{G} = \{\mathcal{S}_n\}_{n=1}^{N_G}$, and a single (evaluation) target set $\mathcal{T}$. A support set is a set of input-label pairs $\mathcal{S} = \{(x_n, y_n)\}_{n=1}^{N_S}$ belonging to a number of different classes. A target set is a set of previously unseen input-output pairs $\mathcal{T} = \{(x_m, y_m)\}_{m=1}^{N_T}$ belonging to the same classes as in $\mathcal{G}$ such that $y_n, y_m \in \mathcal{T}^y \equiv \mathcal{G}^y$.

A CFSL task is controlled by the following hyperparameters:

1. *Number of Support Sets Per Task* (NSS): the cardinality of $\mathcal{G}$, i.e. N_G
2. *Number of Classes Per Support Set* (N_C): the number of unique classes within each support set (way)
3. *Number of Samples Per Support Set Class* (K_S): the number of data-points to be sampled for each class in the support set
4. *Number of Samples Per Target Set Class* (K_T): the number of data-points to be sampled for each class in the target set
5. *Class Change Interval* (CCI): the number of support sets sampled from the same class source, before that class source is resampled
6. *Overwrite* (O): a boolean variable that describes whether classes sampled at each support set will overwrite the class labels of the previously sampled support set (TRUE), or whether they will assigned new unique labels (FALSE)

The process of generating CFSL tasks is described in Algorithm 1. The exact hyperparameter combinations have been left out of this document as we allow freedom in setting arbitrary values to account for the wide research interest. We also strongly encourage researchers to set the hyperparameter values inline with the existing literature.

Algorithm 1 Sampling a Continual Few-Shot Task

```
Data: Given labeled dataset $\mathcal{D}$, number of support sets per
      task $NSS$, number of classes per support set $N_C$,
      number of samples per support set class $K_S$, number
      of samples per class for target set $K_T$, class change
      interval $CCI$, and class overwrite parameter $O$
$a = 1, b = 1$
for $a \leq (NSS/CCI)$ do
    Sample and remove $N_C$ classes from $\mathcal{D}$
    for $b \leq CCI$ do
        $n \leftarrow a \times CCI + b$
        Sample $K_S + K_T$ samples for each of $N_C$ classes
        Build support $\mathcal{S}_n$ with $K_S$ samples per class
        Build target $\mathcal{T}_n$ with $K_T$ samples per class
        if $O = TRUE$ then
            Assign labels $\{1, \ldots, N_C\}$ to the classes
        else
            Assign labels $\{1 + (a-1) \times N_C, \ldots, N_C \times a\}$
              to the classes
        end
        Store sets $\mathcal{S}_n$ and $\mathcal{T}_n$
    end
end
Combine all target sets $\mathcal{T} = \bigcup_{n=1}^{N_G} \mathcal{T}_n$
Return $(\mathcal{S}_{1 \ldots N_G}, \mathcal{T})$
```

2. Data Flow Dynamics

We restrict how a CFSL algorithm is allowed to process data in a given task. The model can only access one support set at a time for the purposes of knowledge extraction. Once this extraction has been completed, the current support set is deleted. Task knowledge can be stored within a parameter vector/matrix or an embedding vector/matrix. Once knowledge has been extracted by all the support sets, the model is tasked with predicting the classes of previously unseen samples in the target set. A generalization measure can be obtained by using the labels of the target set, once the model has produced its predictions to compute a task-level generalization measure.

3. Metrics

We recommend a set of useful metrics to evaluate the methods.

[1] School of Informatics, University of Edinburgh. Correspondence to: Antreas Antoniou <a.antoniou@sms.ed.ac.uk>.

1. Test Generalization Performance: a proposed model should be evaluated on at least the test sets of Omniglot and SlimageNet, on all of the task types of interest. This is done by presenting the model with a number of previously unseen continual tasks sampled from these test sets, and then using the target set metrics as the task-level generalization metrics. To obtain a measure of generalization across the whole test set the model should be evaluated on *600* unique tasks and then take the mean and standard deviation of both the accuracy and cross-entropy performance of the model. These should be used as generalization measures to compare the model to other models.

2. Across-Task Memory (ATM): a proposed model should also report the storage/memory used for any additional information kept from one support to the next. Specifically, a model storing representations of inputs and output vectors, keeps them in its local memory bank $\mathcal{M} = \{(\hat{\mathbf{x}}, \hat{y})_{\mathcal{S}_1}, ..., (\hat{\mathbf{x}}, \hat{y})_{\mathcal{S}_{N_G}}\}$ where $\hat{\mathbf{x}} \in \mathbb{R}^F$ and $\hat{y}$ are representations of the original input vector $\mathbf{x} \in \mathbb{R}^H$ and output y. The amount of compression of input vectors is measured using Across-Task Memory (ATM):

$$\text{ATM} = \frac{|\mathcal{M}^{\hat{x}}|}{|\mathcal{G}^x|}, \tag{1}$$

where $|\mathcal{M}^{\hat{x}}|$ is the max number of stored representation vectors of the support sets at any point during the task (which can include both dynamic or *fast* weights, as well as embedding vectors $\hat{\mathbf{x}}$) and $|\mathcal{G}^x|$ is the total number of samples seen across all support sets ($\mathcal{G}^x = \cup_{n=1}^{N_G} \mathcal{S}_n^x$). To reduce the notation burden we have only considered the inputs $\mathbf{x}$ and not the targets y, since $\mathbf{x}$ is significantly larger than y. We note that $|\mathcal{M}^{\hat{x}}|$ might not be equal to $|\mathcal{G}^x|$, for example when the model is selective about which representations to keep or discard. We also note that for each utilized floating point arithmetic unit we include a computation that takes into account the floating point precision level. For example, if both $\mathcal{M}^{\hat{x}}$ and $\mathcal{G}^x$ use the same floating point standard then it is divided out, but if the representational form uses a lower precision than the actual data-points then it becomes compressive.

3. Multiply-Addition operations (MACs): this metric measures the computational expense of the learner and model operations during learning and inference time. This is different than ATM, as ATM reflects how much memory is required to store information about a support when the next support set is observed, whereas the inference memory footprint measures the memory footprint that the model itself needs to execute during one cycle of inference, and meta-learning cycle.

4. CSFL Rules

1. A task with $CCI = 1$, $NSS = 1$ will generate the same task type no matter what Overwrite is set to.

2. When classes are resampled for a new support set, assuming the CCI interval has been reached, the classes should be unique classes that have not appeared in any of the preceeding support sets within the current continual task.

3. When new samples are sampled, they should be unique samples, that have not been used in any other support set of the current continual task.

4. For SlimageNet the splits should be the exact splits that appear within train, val, and test in https://zenodo.org/record/3672132.

5. The smaller the ATM of a given model the more memory efficient it is. Maximal efficiency is achieved at 0, where no memory is used, and minimum efficiency is reached at infinity, where the model has infinite memory. A model that can store the whole observed dataset will have $ATM = 1$, whereas a model that stores 10 of the information in a data-point will have an ATM=0.1.