Baselines : NGAME

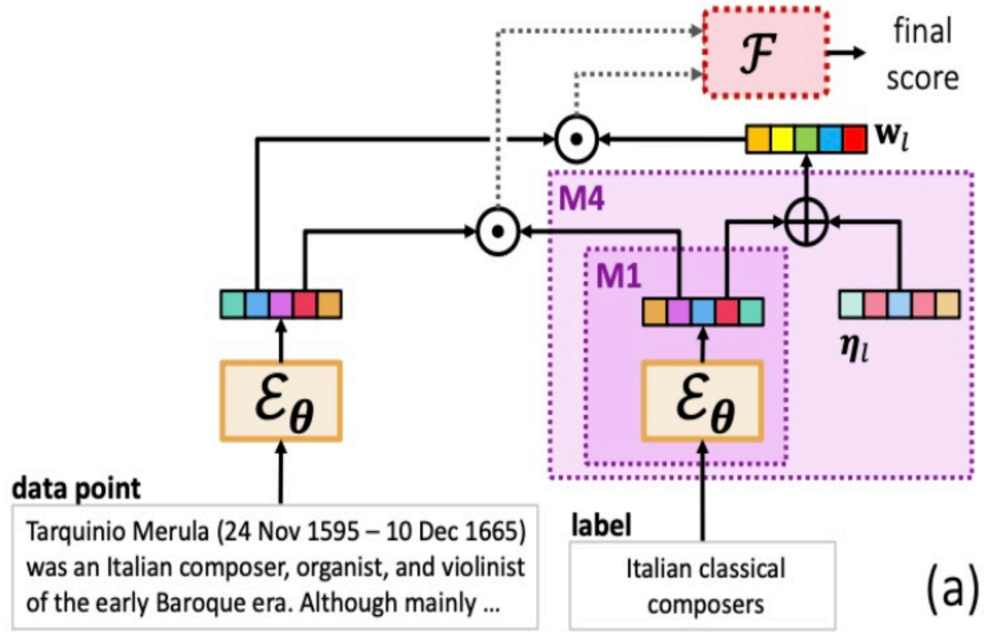
- Proposed lightweight negative in-batch sampling technique for XC
- <<ADD proposed in-batch sampling>>

  – hierarchical KMeans,

- Uses **Curriculum Learning Strategy** - starting with small cluster size & doubling after every 25 epochs
- **Architecture:** DistilBERT-Base Encoder
- Results:
  - NGame yields +8% accurate results & 4x faster to train
  - Adds 1% overhead compared to ACNE 210%
  - Improved Click yield 3%, Coverage 10% & Quality 6%

- Introduced **Modular Training** having 4 phases:
  M1: Encoder Training - uses Siamese Network
  $$\mathbf{w}_l = \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_l).$$
  M2: Negative Mining. Subsumed in M1 and M4

  M3: Feature Transfer - Frozen $\boldsymbol{\theta}$ & Initialize
  $$\mathbf{w}_l = \mathcal{E}_{\hat{\boldsymbol{\theta}}}(\mathbf{z}_l)$$
  M4: Classifier Training - Learn **free parameters** & **W**
  $$\mathbf{w}_l = \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_l) + \boldsymbol{\eta}_l$$



(a)

- **Modular training splits the M1 & M4** training phases, thereby, easing the memory overheads and allowing larger mini-batch sizes
- Leverages **Multi-GPU learning** during M4 (learn W, 1 vs all classifiers) embeddings as embeddings are frozen

Objective Function **M4**:

$$\min_{\boldsymbol{\theta},\mathbf{W}} \mathcal{L}(\boldsymbol{\theta},\mathbf{W}) = \sum_{i=1}^{N} \sum_{\substack{l:y_{il}=+1\\k:y_{ik}=-1}} [\mathbf{w}_k^\top \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{x}_i) - \mathbf{w}_l^\top \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{x}_i) + \gamma]_+$$

Baselines : DEXA

- 2 stage training leads to **distorted encoder** due to the assumption:

$$\mathbf{w}_l \approx \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_l)$$

- **Reason:** Semantic gap between w_l and $\epsilon_\theta(z_l)$
- Leads to poor Encoder training
- DEXA introduced **auxiliary variables (a)** during the encoder training time itself
- Overcomes the increase in # parameters (aux matrix)[***O(L * d)***] by assuming related labels (*l* and *l'*) have same **correction terms** ***a$_{C(l)}$***

$$w_l - \epsilon_\theta(z_l) \approx w_{l'} - \epsilon_\theta(z_{l'})$$

- Uses **Hierarchical Balanced KMeans Clustering** to group the related labels
- Uses DistilBERT-Base encoder and achieves **C <<< *L* .** [***O(C * d)***]

M1: Encoder Training - Learn $\boldsymbol{\theta}$

$$\mathbf{w}_l = \mathfrak{N}(\mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_l) + \mathbf{a}_{C(l)}),$$

Objective:

$$\min_{\{\boldsymbol{\eta}_l\}} \mathcal{L}(\{\boldsymbol{\eta}_l\}) = \sum_{i=1}^{N} \sum_{\substack{l: y_{il}=+1 \\ m \in \hat{\mathcal{N}}_i}} [\mathfrak{N}(\mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_m) + \mathbf{a}_{C(k)})^\top \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{x}_i) - \mathfrak{N}(\mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_l) + \mathbf{a}_{C(l)})^\top \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{x}_i) + \gamma]_+,$$

M2: Classifier Training - Learn $\boldsymbol{w}$

$$\hat{\mathbf{z}}_l \stackrel{\text{def}}{=} \mathfrak{N}(\mathcal{E}_{\boldsymbol{\theta}}(\mathbf{z}_l) + \mathbf{a}_{C(l)})$$

$$\mathbf{w}_l \stackrel{\text{init}}{\longleftarrow} \hat{\mathbf{z}}_l$$

$$\min_{\{\mathbf{w}_l\}} \mathcal{L}(\{\mathbf{w}_l\}) = \sum_{i=1}^{N} \sum_{\substack{l: y_{il}=+1 \\ m \in \hat{\mathcal{N}}_i}} [\mathbf{w}_m^\top \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{x}_i) - \mathbf{w}_l^\top \mathcal{E}_{\boldsymbol{\theta}}(\mathbf{x}_i) + \gamma]_+$$

- **+2.5% - 5.5%** gains over NGame (C=0), +6% accuracy gain on SOTA Deep XC models
- **Minimal** overhead on training time & **no overhead** on model size