A Experiments Details

A.1 Baseline Models

- **SVM** (Lv et al., 2015): A support vector machine with a linear kernel (c=10) takes the proportion of suicide lexicon occurrence in a post as input.
- **Random Forest (RF)** (Amini et al., 2016): A Random Forest takes the proportion of suicide lexicon occurrence in a post as input.
- **Contextual CNN (C-CNN)** (Gaur et al., 2019): The C-CNN (Shin et al., 2018) is trained with posts which are encoded by ConceptNet word embeddings (Speer and Lowry-Duda, 2017).
- **SISMO** (Sawhney et al., 2021a): The SISMO uses Longformer (Beltagy et al., 2020) to obtain post embeddings. The extracted embeddings are fed into the Bidirectional LSTM. Unlike previous work (Sawhney et al., 2021a), we apply this model for post-level predictions.
- **BERT**: We use the BERT-BASE (Devlin et al., 2018) model, and most hyperparameters are chosen from the original paper.
- **Suicide Detection Model (SDM)** (Cao et al., 2019): Following the SDM, we fine-tune the word embedding model by adopting the masking method to capture domain knowledge from a pre-built suicide dictionary. The obtained embeddings from fine-tuned embedding model are fed into an LSTM model with an attention mechanism.
- **Reformed BERT**: We recreate the pre-trained BERT tokenizer (Devlin et al., 2018) by adding the suicide lexicons.

A.2 Experimental Setup

We tune hyperparameters based on the highest FScore obtained from the validation set for all the models. We use the grid search to explore (i) number of kernel output size in aggregate function $\tilde{q} \in \{50, 60, 70, 100\}$, (ii) number of post features in hidden state $\tilde{H^D} \in \{64, 128, 256, 512\}$, (iii) initial learning rate $lr \in \{0.01, 0.001, 2e{-}5, 3e{-}5, 5e{-}5\}$, and (iv) dropout rate $\sigma \in \{0.0, 0.1, ..., 0.5\}$. The optimal hyperparameters were found to be: $\tilde{q} = 50, \tilde{H^D} = 512, lr = 3e - 5$ and $\sigma = 0.1$. We implement all the methods using PyTorch 1.6 and optimize with the mini-batch AdamW (Loshchilov and Hutter, 2017) with a batch size of 32 and lr = $3e - 5$. We use the Exponential Learning rate Scheduler with gamma 0.5. Following Hamilton et al. (2017), we choose the maximal search depth k as 2, and 100,140 neighbors sampling for k = 1, 2 respectively. We train the model on a GeForce RTX 2080 Ti GPU for 20 epochs and apply early stopping with a patience of 7 epochs.