# ALTEGRAD Challenge 2023

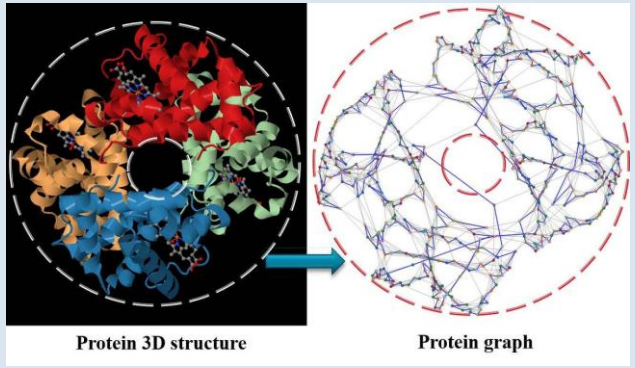
Name : Tom Salembien

Kaggle Name : Ohm_Tom

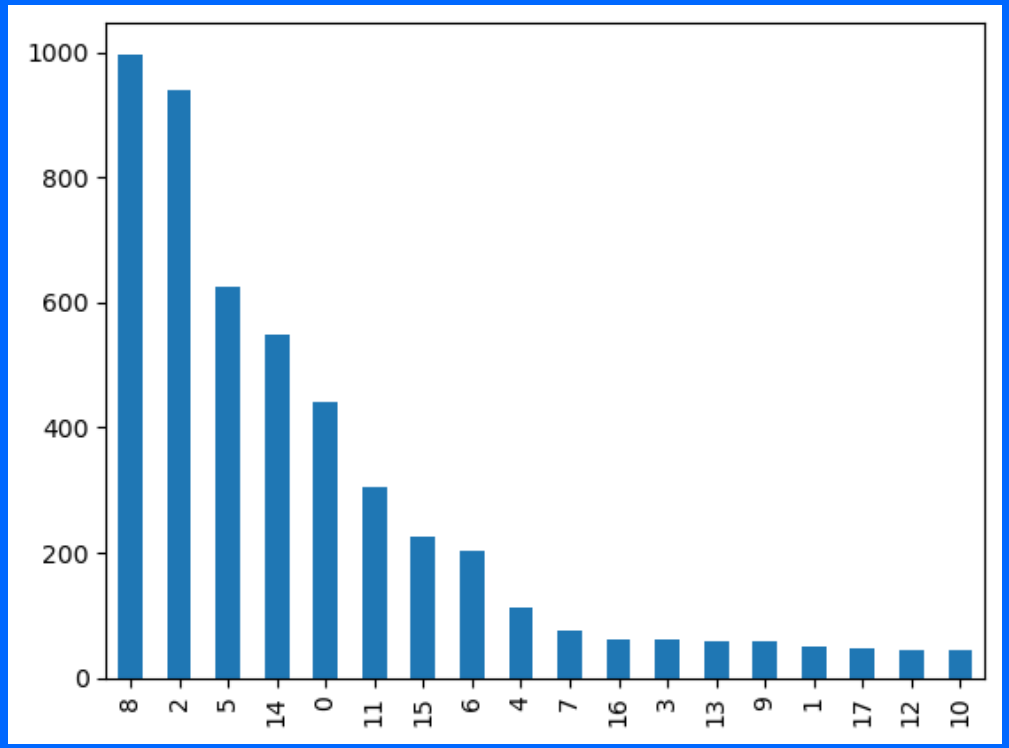
# Summary

1. Introduction
2. Baseline Approach
3. CNN_BiLSTM
4. HGP_SL model
5. Tries and Fails

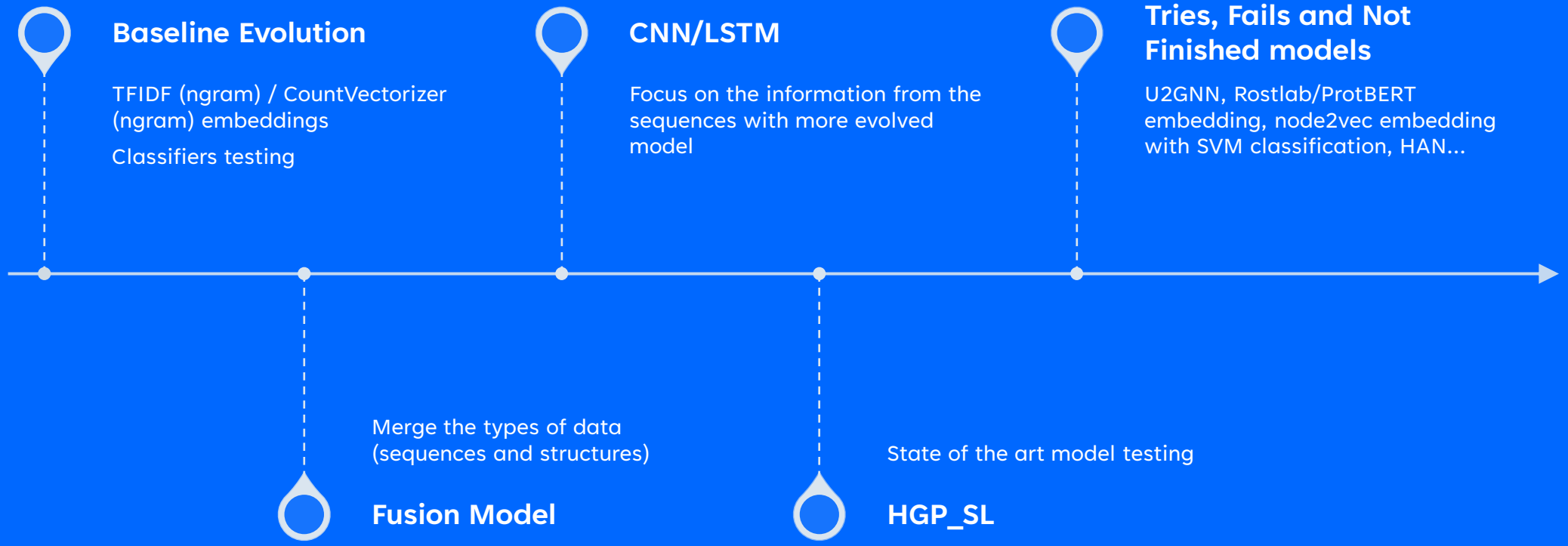
# Introduction




- Goal
- Dataset
- Complexity

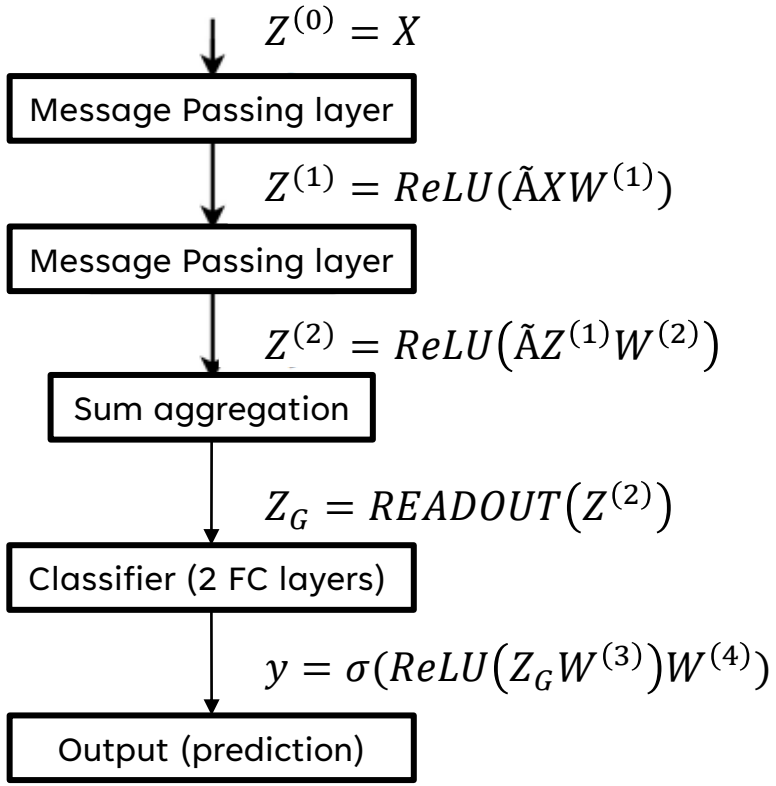


# Overview Approaches



# Baseline Evolution

**Structure : Stack message passing layers**




**Example GNN for 3 graphs**

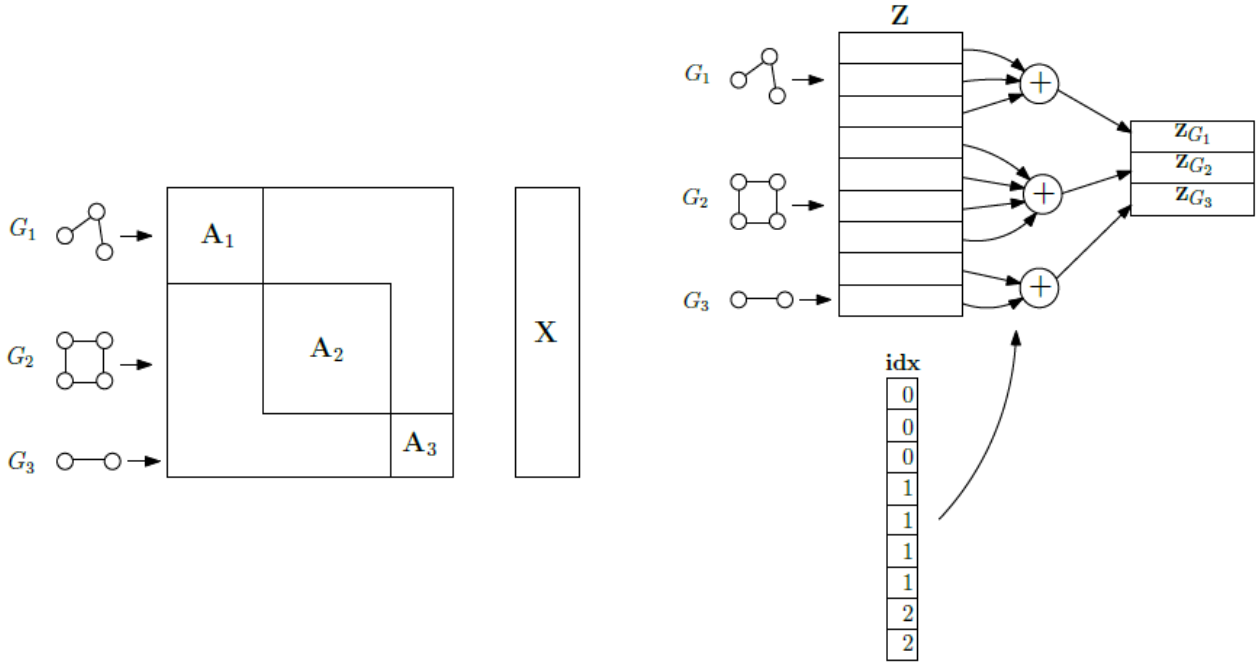



Figure 2: Implementation details of the graph neural network.

# Limits of stacking GNN layers

- Problems of Oversmoothing : all node embeddings converge to the same value
- Bad as we want to use node embeddings to differentiates the graphs
- Why ? Because of the receptive field (set of nodes that determine the embedding of a node of interest), In a kth message passing layer, each node has a receptive field of k-hop neighborhood -> fast cover of all graph (shared neighbors grows)
- Many message passing layers -> nodes highly-overlapped receptive fields -> their embeddings are highly similar -> over smoothing

# Sequence first analyses

## Sequence : Embedding + Classification

Embeddings,

- TF-IDF with different n_grams
- Count Vectorizer with different n_grams

Then classifiers,

- Logistic Regression (simple, elasticnet with saga)
- Tree based models (Random Forest)
- MLP

## TFIDF embedding

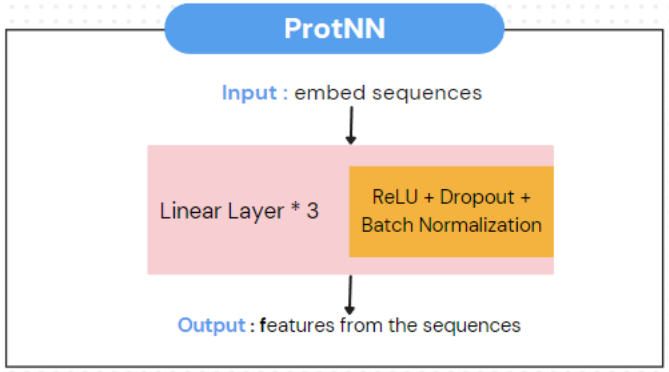
$$\text{tfidf}_{i,j} = \text{tf}_{i,j} \times \log\left(\frac{\text{N}}{\text{df}_i}\right)$$

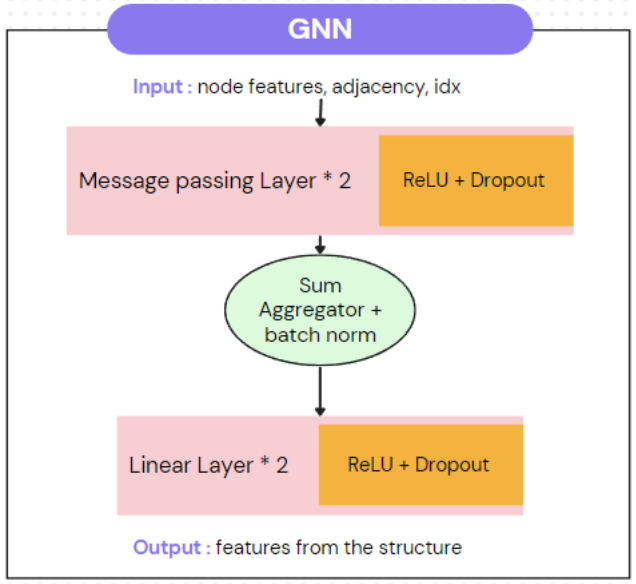
$\text{tf}_{i,j}$ = total number of occurences of i in j

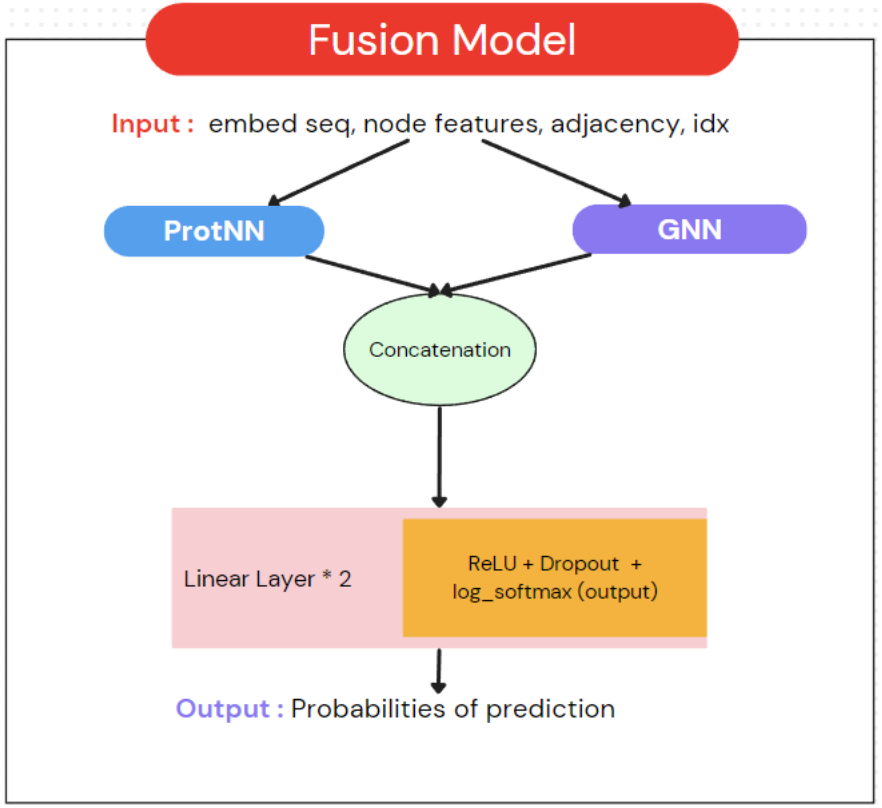
$\text{df}_i$ = total number of documents (speeches) containing i

N = total number of documents (speeches)

# Fusion Model



# CNN+BiLSTM

## Embedding :

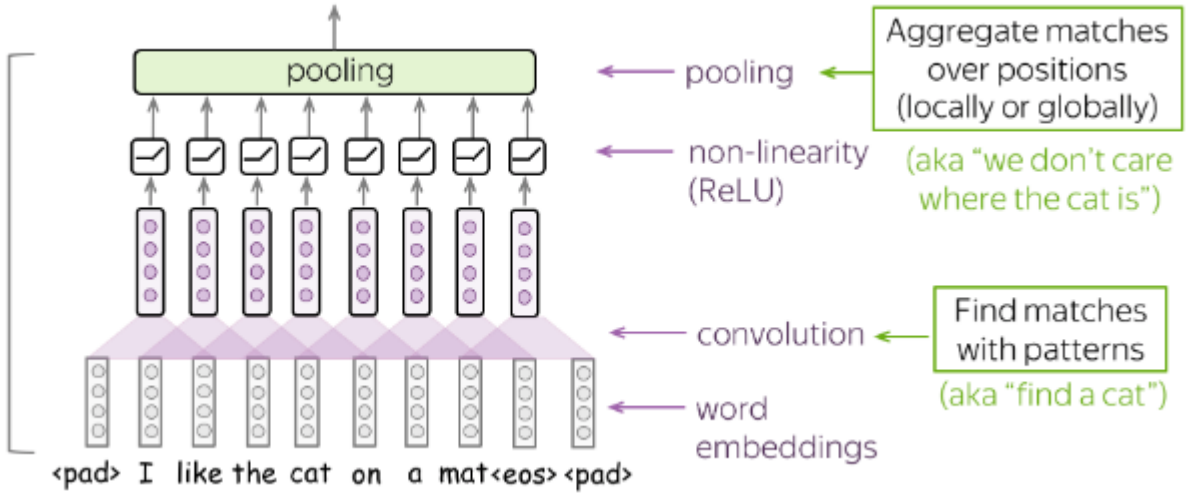
- Tokenizer from keras :

Fit_on_texts : creates the vocabulary index based on amino acid frequency (lower integer means more frequent amino acid)

Texts_to_sequences : takes each character in the text and replaces it with its corresponding integer value from the built dictionary

Padding and truncation at 600 characters

## CNN for texts mining :

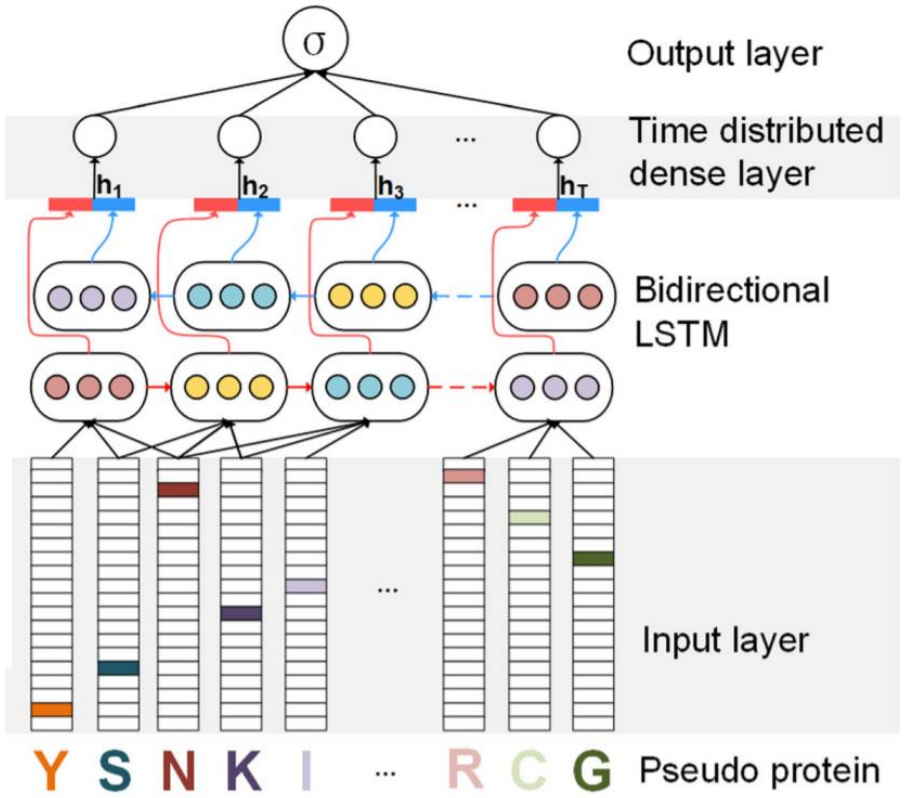



Similar with proteins

Kernel_size(`[3, 10, 20, 40]*21, 300 filters`), padding(n

Stride (1,1) ...

## BiLSTM




## My Model

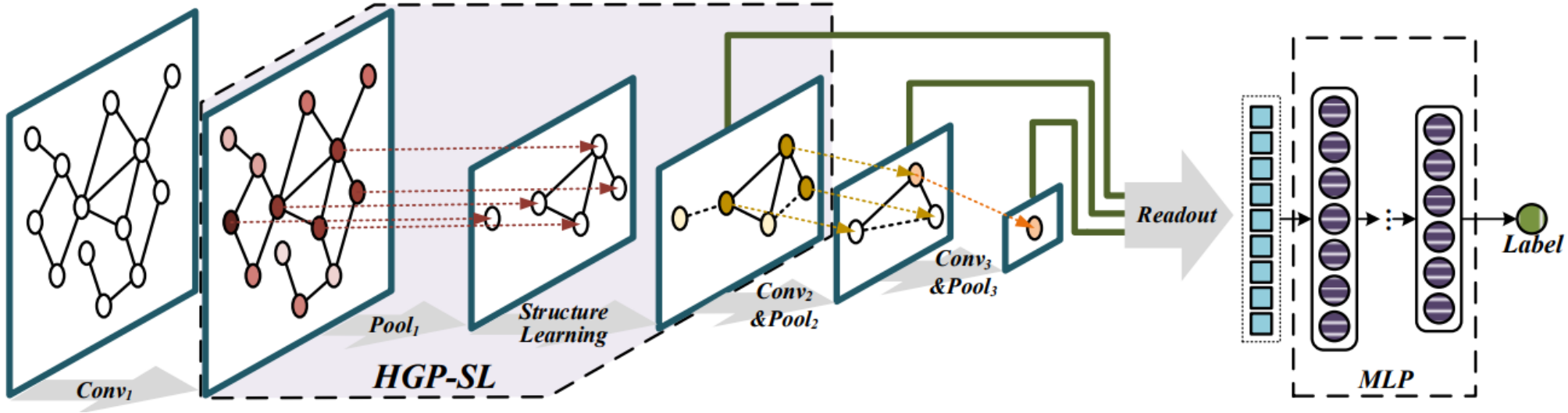
Similar to the Fusion model, with extraction of the features with on the first hand, Convolution+pooling and concatenation (with dropout).

One the other hand, concatenation of the features extracted by the BiLSTM, the *output* contains a concatenation of the forward and reverse hidden states at each time step in the sequence.

Finally, we concatenate alland classify it with 2 FC layers and log_softmax output.

# HGP_SL

## Graph classification



# Structure and idea of the HGP_SL :

1

## Convolution

Classical graph convolutionnal neural network

2

## Pooling

Special hierarchical pooling of the nodes in the graphs

3

## Structure Learning

Learning of sparse graph structure with sparse attention mechanism

4

## Readout

Sum features extracted from each HGP_SL layers (3 in our model)

5

## MLP

Classifaction layers

# Convolution : GCN

For the graph i, in the kth layer of a GCN :

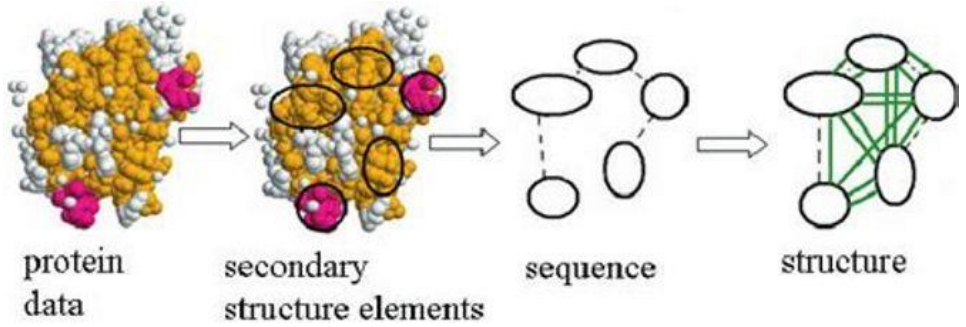
$$H_0 = X \text{ and } \tilde{A} = A + I \text{ and } \widetilde{D} \text{ degree matrix of } \tilde{A}$$

$$\boldsymbol{H_{k+1} = \sigma(\widetilde{D}^{\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H_k W^k)}$$

With output dimension of weight matrix equal for all layers for simplicity.

Here, we have normalized A so that the feature vectors scale doesn't change, we do that degree matrix $D^{-1}A$, in practice the dynamics get more interresting with symetric normalization thus we have the formula.

# Pooling




1. Preserves a subset of informative nodes and form smaller inner subgraph
2. Non parametric pooling operation that can utilize both node features and graph structure information
3. Use criterion (node information score) that guides node selection procedure
4. **Node information score** :if a node's representation can be reconstructed by its neighborhood representations, it means this node can probably be deleted in the pooled graph with almost no information loss
5. Manhattan distance between the node representation itself and the one constructed from its neighbors : $\text{score}(\text{graph}_\text{i}) = p = ||\left(I_i^k - \left(D_i^k\right)^{-1} A_i^k\right) H_i^k||_1$, l1_norm row-wisely.
6. Reorder nodes in graph according to their information score
7. Creation of top ranked nodes (take pooling_ratio*number of nodes in graph) and pool :

**Node ftrs :** $H_i^{k+1} = H_i^k(idx, :)$ , **Graph_str_info** : $A_i^{k+1} = A_i^k(idx, idx)$

# Structure Learning

- Pooling -> highly related nodes are disconnected ->lost of completeness of graph structure information
- To learn the refined graph structure that encodes the underlying pairwise relationship, they use a single FC layer with then a similarity score between the nodes of the pair with attention mechanism :

$$\mathbf{E}_i^k(p,q) = \sigma(\vec{\mathbf{a}}\ [\mathbf{H}_i^k(p,:)||\mathbf{H}_i^k(q,:)]^\top) + \lambda \cdot \mathbf{A}_i^k(p,q)$$

$A_i^k$ *encodes the induces subgraph structure information*

- To bias the attention mechanism to give larger score to directly connected nodes in the graph and try to learn the relations between disconnected nodes
- To compare these scores we need metric : sparsemax (not softmax because non zero values -> dense FC layer in SL so noisy) to retain most important properties of softmax fct while producing sparse distributions:

$$\mathbf{S}_i^k(p,q) = \text{sparsemax}(\mathbf{E}_i^k(p,q))$$

$$\text{sparsemax}(\mathbf{E}_i^k(p,q)) = [\mathbf{E}_i^k(p,q) - \tau(\mathbf{E}_i^k(p,:))]_+$$

# Readout / MLP Classification

- To get fixed size graph level representation -> use readout function that aggregates all nodes representation in the subgraph $H_i^1, H_i^2, \ldots, H_i^K$. Use concat of mean-pooling and max-pooling :

$$\mathbf{r}_i^k = \mathcal{R}(\mathbf{H}_i^k) = \sigma\left(\frac{1}{n_i^k}\sum_{p=1}^{n_i^k}\mathbf{H}_i^k(p,:)\,||\,\max_{q=1}^{d}\mathbf{H}_i^k(:,q)\right)$$

- And finally the representation looks like (Z summarize the differents graph level representation):

$$\hat{\mathbf{Y}} = \mathrm{softmax}(\mathrm{MLP}(\mathbf{Z})) \quad with\ z_i = r_i^1 + r_i^2 + \ \ldots + r_i^K$$

$$\mathcal{L} = -\sum_{i\in L}\sum_{j=1}^{c}\mathbf{Y}_{ij}\log\hat{\mathbf{Y}}_{ij}$$

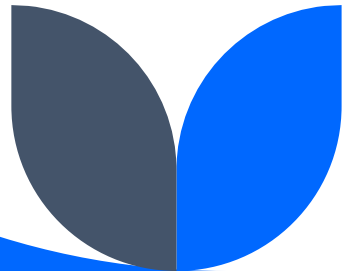
# Results and Discussion

| | Loss | Time |
|---|---|---|
| HGP_SL | 1.86 | Very Long |
| CNN+BiLSTM | 1.61 | Normal/Long |
| Embedding + MLP | 1.38 | Normal |
| Fusion model | 1.52 | Long |
| Embedding + Logistic regression | 1.26 | Fast |

# Tries and Fails

- ProtBERT from hugging face
- Try enhance baseline model
- Directly state of the art model –> wrong
- Data augmentation / pb unbalanced dataset

- Methodology in working process / workflow
- Lost of time with training of HGP_SL
- Should focus more on features extraction



# Thank you !

## References :


[Kipf and Welling 2017] Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. ICLR.

[Lee, Lee, and Kang 2019] Lee, J.; Lee, I.; and Kang, J. 2019. Self-attention graph pooling. In ICML, 3734–3743.

[Li et al. 2018] Li, R.; Wang, S.; Zhu, F.; and Huang, J. 2018. Adaptive graph convolutional neural networks. In AAAI.

[Ma et al. 2019] Ma, Y.; Wang, S.; Aggarwal, C. C. and Tang, J. 2019. Graph convolutional networks with eigenpooling. In SIGKDD.

[Martins and Astudillo 2016] Martins, A., and Astudillo, R. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In ICML, 1614–1623.

[Hongyang Gao2 February 2021] Graph Neural Networks: A Feature and Structure Learning Approach

[Zhen Zhang, Jiajun Bu, Martin Ester, Jianfeng Zhang, Chengwei Yao, Zhi Yu, CanWang, 2019] Hierarchical Graph Pooling with Structure Learning

# Annexe Calculus of threshold

**Properties of sparsemax** : non negative and sum to one

**Algorithm 1** The calculation procedure of function $\tau(\cdot)$

**Require:** input vector $\mathbf{z} \in \mathbb{R}^n$.

1: Sort $\mathbf{z}$ into $\mathbf{u}$: $u_1 \geq u_2 \geq \cdots \geq u_n$.

2: Get $\rho = \max\{1 \leq j \leq n : u_j + \frac{1}{j}(1 - \sum_{i=1}^{j} u_i) > 0\}$.

3: Define $\tau(\mathbf{z}) = \frac{1}{\rho}(\sum_{i=1}^{\rho} u_i - 1)$.