**GRASP (Graph And SMILES Pre-training): A Cross-Modal Approach to Self-Supervised Molecular Representation Learning**

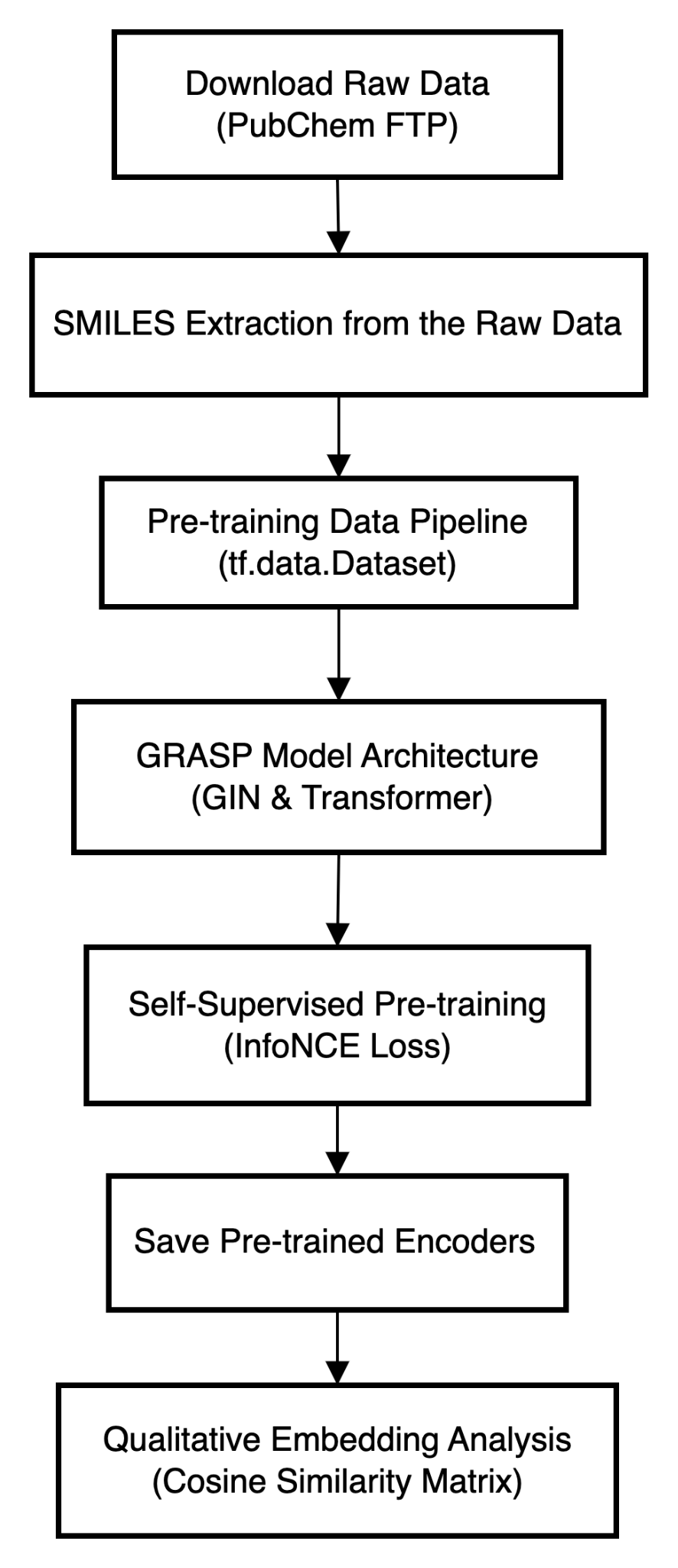
**Summary:**

There is a prevailing bottleneck in Computational Chemistry i.e. very less amount of labelled molecular data is available. Labelling molecular data is both expensive and slow. In our project we plan to build a model that will be able to learn molecular representations by aligning two modalities: **Graph-based structures and SMILE strings,** using **cross-modal contrastive learning**. By using this we aim to learn good, unified molecular embeddings that will capture structural and sequential information. These embeddings can be further fine tuned for tasks such as toxicity prediction, solubility estimation and blood-brain barrier permeability classifications.

**Goals:**

1. We want to train a model that will understand both molecular graphs (via Graph Neural Network) and SMILES strings ( via Transformers).
2. Learn from unlabeled data based on GRASP.
3. **Benchmark Testing:** Evaluate the model on downstream prediction tasks (from MoleculeNet).
4. **Efficiency Proof:** We wish to clearly show that the model performs well with much less labeled data than supervised models.

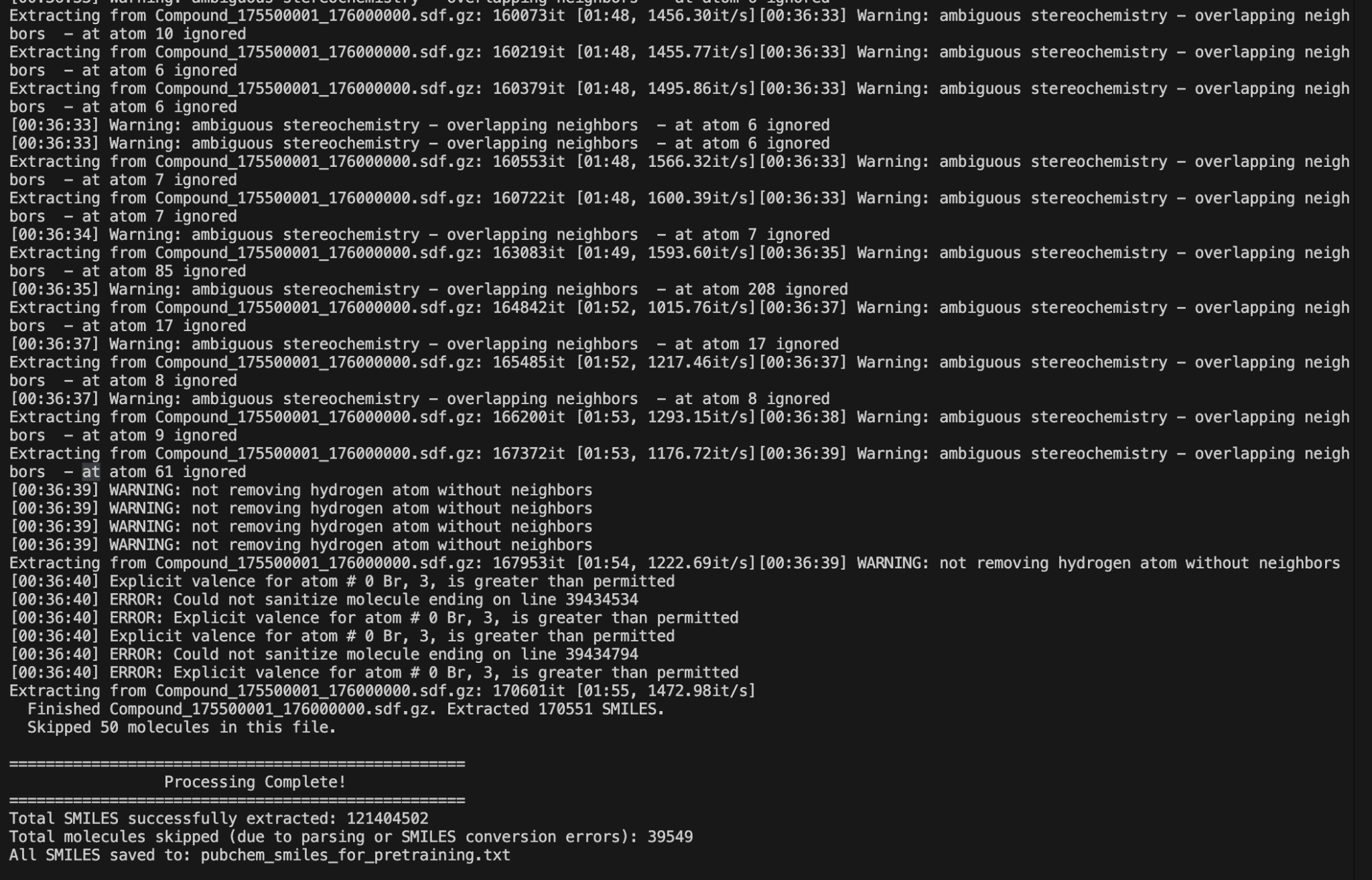
### **Methodology**

Our methodology is grounded in cross-modal contrastive learning, leveraging two distinct encoders to process diverse molecular data modalities. The progress to this checkpoint primarily encompasses data acquisition, preprocessing, and the successful implementation and initiation of the pre-training pipeline.

#### **Data Acquisition and Preprocessing**

The Grasp model relies on a massive unlabeled molecular data for self-supervised pre-training.

* **Data Sourcing:** We have utilised the PubChem database (<https://pubchem.ncbi.nlm.nih.gov/docs/compounds>) . It is a comprehensive repository of chemical information, this was the major source for us of getting unlabeled molecules.
* **Bulk Data Download:** We initiated the download of approx. 352 gzipped SDF format files. These were downloaded from the PubChem FTP server.The downloaded files totalled around 113 GB in compressed form.

****

<https://buffalo.app.box.com/folder/327329701397>

* **Memory-Efficient SMILES Extraction:**  We created a custom python script to process the large volume of compressed .sdf files. Our script implemented an iterative and memory-conservative strategy: Using our script.
  + We open every.[sdf.gz](http://sdf.gz) file and read directly from its compressed state.
  + We installed RDKit to parse each molecule with the .sdf format and extract its canonical SMILES string.
  + The extracted SMILES strings are added to a single, consolidated output text file (pubchem\_smiles\_for\_pretraining.txt), it substantially reduces the size.
  + We deleted the .[sdf.gz](http://sdf.gz) file from disk after it was successfully deleted. This in particular was a crucial step to ensure we have memory available and the system does not face any memory related issues.
  + There are tens of millions of SMILES strings extracted for pre-training by our script.

#### **Model Architecture and Pre-training Setup**

The core of the GRASP model is made of dual-branch based NN architecture which is made suitable for cross-modal understanding.

* **Framework Selection:** TensorFlow 2.x and Keras are used as the primary deep learning framework.
* **Molecular Featurization Pipeline:** we used tf.data.dataset to create the pipeline to prepare molecular data for the neural networks. This pipeline performs the following:
  + **SMILES Tokenization:** The Raw SMILES strings (which are just molecules written in text ) get turned into numerical IDs. which is suitable for input to the Transformer encoder. A character level vocab is also built from the preprocessed dataset like a dictionary for character.
  + **Graph Conversion:** We are using RDKit within **tf.py\_function** to turn SMILES strings into graph representations.Each graph is featured into atom properties as a node and bond connectivity between them as edges.
  + **Dynamic Padding and Batching:** We can have variable size of molecular graphs and SMILES strings, taking it into consideration our pipeline dynamically pads inputs to MAX\_NODES and MAX\_SMILES\_LEN.It flattens out all the edges (connections) from all molecules at once and then applying global node offsets, this just makes sure everything works smoothly with the GIN part, which is pretty picky about how it sees graph data, especially when it's sparse
  + **Efficient Data Loading:** tf.data.Dataset utilities (e.g., cache(), shuffle(), padded\_batch(), prefetch(tf.data.AUTOTUNE)) are employed to optimize data throughput, minimizing bottlenecks during training.
* **GRASP Model Implementation:** Our model's architecture was implemented using custom Keras Model and Layer subclasses:
  + **GIN Layer (GINLayer):** A custom keras layer where we implemented the Graph Isomorphism and Network’s aggregation mechanism which is designed to work with flattened node features and global sparse adjacency matrices.
  + **GIN Encoder (GINEncoder):** This stacks several GINLayers one on top of the other,preceded by an initial MLP. First, it takes the raw atom info to the desired hidden\_dem, and interleaved with Batch Normalization layers to process it and keep things stable. Finally, a global sum pooling operation combines node embeddings into a single graph-level embedding.
  + **Transformer Encoder:** This is implemented with standard Keras MultiHeadAttention and Dense Layers.These are responsible for processing tokenized SMILES sequences. It also includes the learnable positional embeddings and handles the padding via an additive attention mask. We have used a masked global mean pooling to obtain a single SMILES embedding.
  + **Projection Heads :** Simple dense networks that map the higher-dimensional outputs of the GIN and Transformer encoders into a shared, lower-dimensional project\_dim embedding space.
  + **GRASPModel:** It receives raw data tensors, preprocesses edges and nodes dynamically and feeds it to the respective encoders and the projection heads. It also apply the L2 normalization to the final project embeddings.
* **Pre-training Objective (InfoNCE Loss):** The Information Noise-Contrastive Estimation loss function encourages the model to pull graph and SMILES embeddings of the same molecule closer in the shared embedding space. But if they are from different molecules it pushes them apart. This self-supervising signal drives the learning of meaningful representations without explicit labels.
* **Hardware Setup (M1 Metal):** For all this pre-training work, we specifically set up our pipeline optimized for M1 Pro chip. Having gone through the cycle of selecting which gpu to run it on we were familiar with this. So we used the MPS with tensorflow to make sure it utilises M1 chip power.

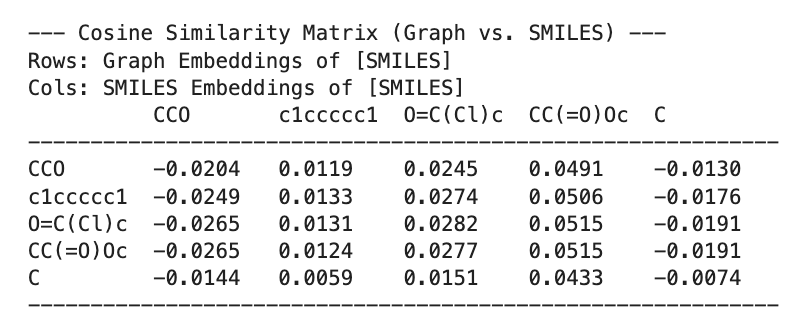
### **Evaluation:**

We have trained a subset of the 100,000 PubChem SMILES strings using the GRASP model and tf.distribute.OneDeviceStrategy on the Apple M1 Pro chip. The model is configured to train for 5 epochs with a learning rate of 1e-4 and a batch size of 64.

* **Current Training Status:** The pre-training on the 100,000-sample subset has been **successfully completed.**
* **Observed Loss (Average InfoNCE Loss per Epoch):**
  + Epoch 1: 1.2832
  + Epoch 2: 1.1770
  + Epoch 3: 1.2637
  + Epoch 4: 1.4348
  + Epoch 5: 1.5458
* **Pre-training Progress:** The model has successfully completed **5** out of 5 epochs. Each epoch consisted of 3125 batches.
* **Training Time:** The pre-training on the 100,000-sample subset took approximately 1 hour and 54 minutes. We anticipate that pre-training on the full PubChem dataset (tens of millions of molecules) will require significantly extended training time, likely several days. We have tried to train our model on different machines (GPUs and TPUs) and a lot of time was devoted to training the model with various subsets of the data. Training of models is still going on for 1 million samples.

#### **Initial Qualitative Embedding Analysis**

We have performed qualitative analysis by computing the cosine similarity between graph and SMILES embeddings for a small set of diverse test molecules. This is done to assess the preliminary quality of the learned molecular representations of the pre-training phase. Our analysis utilised the encoders that are pre-trained on the 100,000-sample subset.

****

**Observations:** After looking at the generated cosine similarity matrix for a selection  
of test molecules, we understand that the values across the diagonals(positive pairs) and off-diagonal (negative pairs) elements are generally close to zero in addition to some slight positive and negative values. This infers that the we have not yet prominently established a strong alignment (diagonal elements i.e. representing graph and SMILES embeddings of the same molecule, are significantly higher than off-diagonal elements i.e. representing different molecules) For example, the similarity between graph(CCO) and SMILES(CCO) is -0.0204, which is not distinctively higher than other similarities in the row.

* **Interpretation:** We can imply that our model’s pre-training on the 100,000 sample subset has not fully converged to a state where it can consistently learn robust and separable cross-modal embeddings for individual molecules. It highlights that we are going to need to continue our training on a larger subset of the dataset.
* **Next Steps for Pre-training:**
  + Running the pre-training for its full duration (5 epochs or more) **on a larger subset of the PubChem dataset**.
  + Monitoring the InfoNCE loss for further convergence throughout the extended training for above.
  + Post-checkpoint, further hyperparameter tuning is required for the InfoNCE loss, particularly the temperature parameter, and which can potentially increase the total number of pre-training epochs, will be critical to achieve the desired alignment.

### **4. Plan for Final Submission**

### 

For the checkpoint, our evaluation focuses on the operational status of the pre-training pipeline. Comprehensive quantitative evaluation metrics and visualizations will be presented in the final project submission.

### **5. Dataset**

The project primarily utilizes two categories of datasets:

* **Unlabeled Pre-training Dataset(have used this till now):**We have utilised the PubChem database (<https://pubchem.ncbi.nlm.nih.gov/docs/compounds>) . It is a comprehensive repository of chemical information, this was the major source for us of getting unlabeled molecules. We initiated the download of approx. 352 gzipped SDF format files. These were downloaded from the PubChem FTP server.The downloaded files totalled around 113 GB in compressed form.
* **Labeled Downstream Benchmarking Datasets(will use this for the next phase):** For evaluating the learned representations, we will use datasets from the MoleculeNet benchmark. Specifically, we will leverage:
  + **BBBP (Blood-Brain Barrier Permeability):** To predict if a compound can cross the blood-brain barrier.
  + **Tox21:** To predict the toxicity of compounds based on various assays.
  + **ESOL:** For predicting the water solubility of compounds. These datasets provide SMILES strings with associated experimental labels.

**References:**

1. **Contrastive Learning:** Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). *A Simple Framework for Contrastive Learning of Visual Representations (SimCLR)*. ICML.
2. **Cross-Modal Learning:** Radford, A., et al. (2021). *Learning Transferable Visual Models From Natural Language Supervision (CLIP)*. ICML.
3. **Graph Neural Networks:** Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). *How Powerful are Graph Neural Networks?*. ICLR.
4. **Transformers for Chemistry:** Chithrananda, S., Grand, G., & Ramsundar, B. (2020). *ChemBERTa: Large-Scale Self-Supervised Pre Training for Molecular Property Prediction*. arXiv.
5. **Molecular Benchmarking:** Wu, Z., et al. (2018). *MoleculeNet: A Benchmark for Molecular Machine Learning*. Chemical Science.