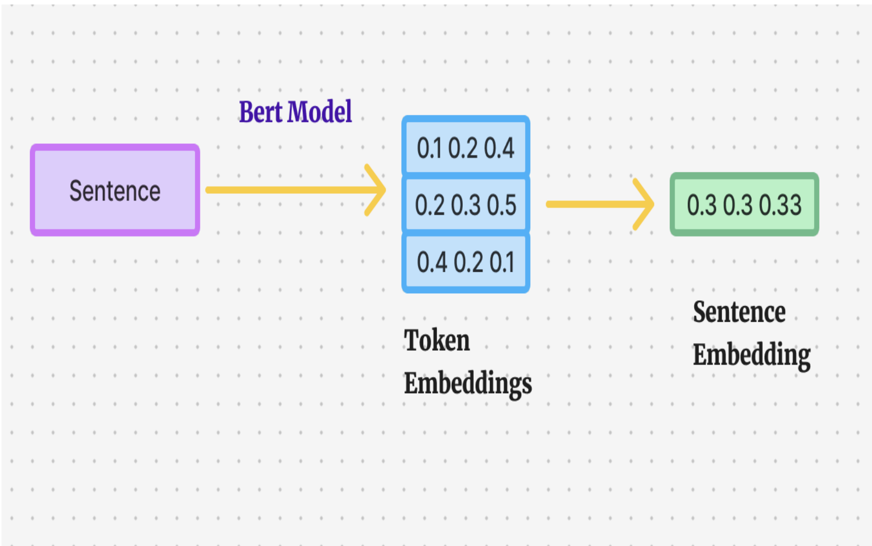
Task 1:



Model Configuration :

- ‘Bert-base-uncased’ as backbone.
- Configurable embedding size.

Pooling strategy:

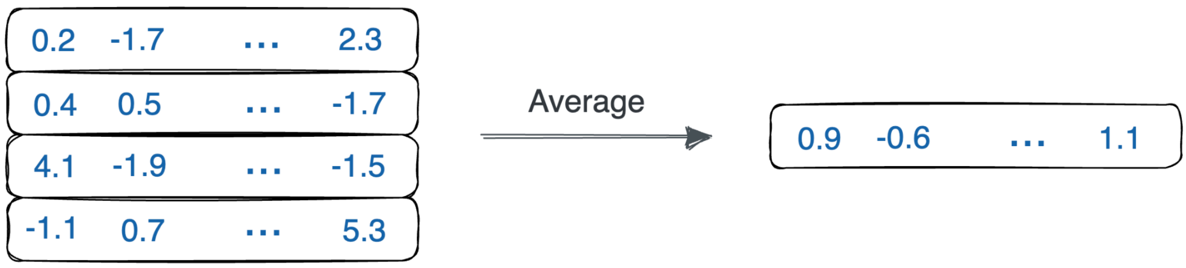
- Implemented three different pooling strategies
    1. Mean : averages pooling across all token embeddings.
    2. Cls : Using [Cls] token out from pooler_output.
    3. Max : max pooling across token embeddings.

Embeddings Normalization:

- Optional embeddings normalization.
- Used L1-norm.

❖ Using Bert we get sequence of token embeddings, instead of single sentence-level embeddings. To get the latter, we need to perform post processing technique like pooling.

❖ E.g.: Avg pooling - averages pooling across all token embeddings.



Pitfall:

- Mean or average pooling results in inevitably loss of nuanced information which is captured by word-level embeddings. In other words, contextual information dilution into a squashed single vector that the Bert model captures.

Workaround:

- **Use Sentence encoders**

  1. **Cross-encoder**

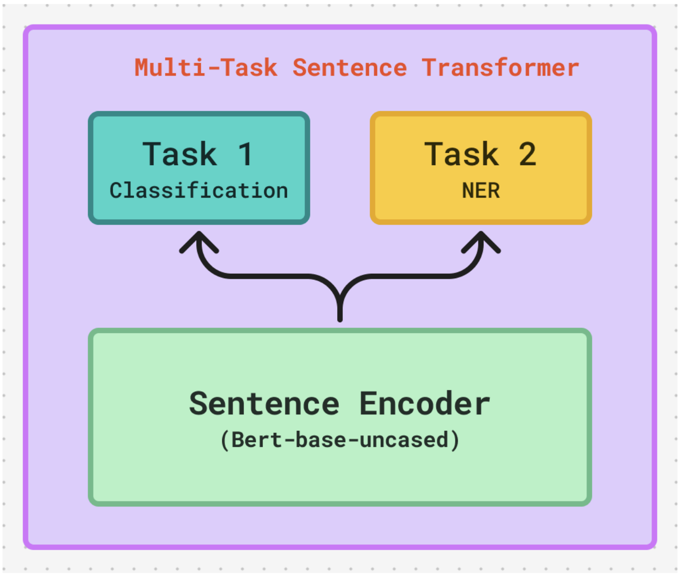
  Here two sentences are concatenated before passing them to BERT model. Train the Bert model as you normally would and take the output corresponding to [CLS] token.

  2. **Bi-encoder**

- Embed inputs independently.
- Create sentence embedding using post-processing like pooling.
- Train bi-encoder : once we get both the embeddings, compare them to find similarity.

Task 2

❖ **Modifications for Multi-task Learning:**



1. **Shared component - bert-base-uncased:**
- Bert base transformer remains common.
- Shared tokenizer.
- Common hidden size from base transformer.
2. **Task Specific heads:**
   a. **Classification head:** performs hidden_size projection to num_classes.
   b. **Named Entity Recognition head:** performs hidden_size projection to num_ner_labels.

Head Architecture:

- Both heads follow same structure i.e. dropout for regularization, layer norm, GELU, task specific final projection.

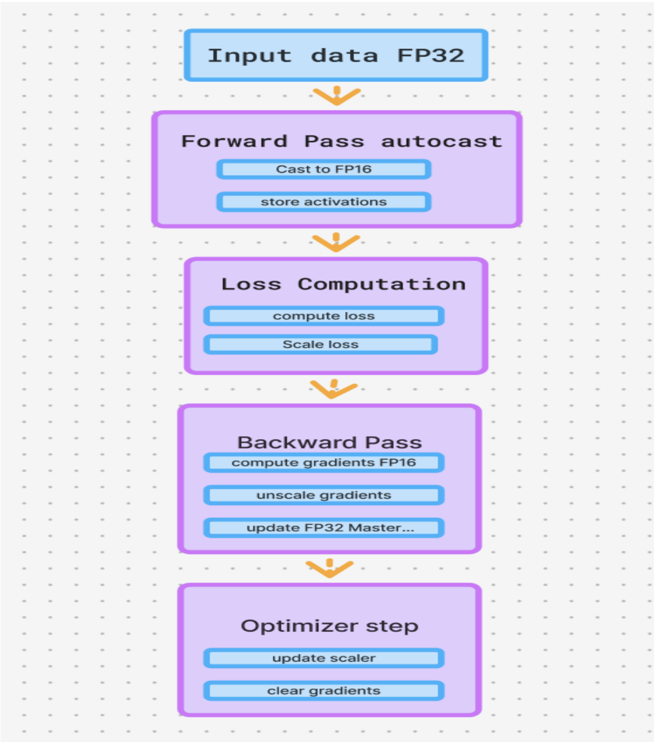
❖ **Dataset used for Multi-task Learning:**

1. **Sentence Classification :**
   <u>AG News Dataset:</u>
   4 Classes : World News, Sports, Business, Science/Tech.
   Features : Title, Description.
   - Single label per text.
   Num Samples : Train - 120k, Test 7.6k
2. **Named Entity Recognition:**
   <u>CoNLL2003 Dataset:</u>
   4 Entity types: PER, ORG, LOC, MISC

Num Samples: Train - 3.25k, Test 3.2k
Final Dataset training size = Minimum(AG news data size, CoNLL data size)

- Utilized PyTorch Datacollator to prepare task-specific data batches for model input and handling consistent formatting and padding.

❖ **Mixed Precision Training:**



Utilized torch.cuda autocast function and GradScaler() for mixed precision training.

Process :
- Use 16bit Float for forward pass, activation storage & most computations.
- Keep 32bit Float for model params, final gradients.

Benefits :
- Reduced memory usage
- Faster training.

Observation :
Observed training time reduced by 2.5x as compared to w/o mixed precision training.

Advantages of Multi-task architecture:
1. Shared feature learning.
2. Efficient parameter usage.

TASK 3

1. **Freezing Entire network:**
- Results in no parameter updates during training and fixed feature extraction.
- Advantage is memory efficient and fast inference.
- Disadvantage is no adaptation to new data and cannot perform task-specific optimization.
2. **Freezing Transformer backbone:**
- Partial adaptation and fixed feature extraction. Although allows trainable task-specific layers.
- Advantage is it preserves pretrained knowledge, faster training and ideal for limited data setting.
- Disadvantage is it may not optimally capture task-specific features.
3. **Freezing one task head:**
- Allows full feature adaptation, task-specific knowledge preservation and mixed task optimization.
- Advantages are it optimized for specific task while maintaining performance on frozen task. Moreover, results in balanced resource usage.
- Disadvantages are potential task interference and complex optimization scenario.

Recommended approach:
- Use adaptive training
1. For small dataset: freeze backbone.
2. Large dataset: Train entire model(Full model adaptation).
3. Similar tasks: Use shared training with knowledge sharing or use task specific training.
   Use task-specific learning rates.

Transfer Learning Considerations
1. **Choice of pre-trained model:**
- Bert-base-uncased : General purpose and well-suited for English texts. Good balance of performance and size.
- RoBERTa : Better performing than BERT but requires more computing. Ideal for performance-critical tasks.
- DistilBERT : Uses knowledge distillation hence lighter & faster. Ideal for limited computing & memory resources.

Rationale: identify data size, task requirements and available compute to decide right pre-trained model backbone.

2. **Choice of layer Freezing:**

- Full Freezing: Freezing entire base model and train task heads. Good for small dataset and it prevents overfitting.
- Gradual Unfreezing: Start with frozen base model and train task heads initially. Gradually unfreeze from the top-down approach.
- Selective Freezing: Keeping lower layers frozen(which captures general features). Unfreeze task-specific upper layers. Train task heads always.

Rationale:

- Small dataset: freeze more layers and focus on task head training.
- Large Dataset: Unfreeze more layers and allowing model adaptation.

Task similarity:

- Similar to pre-training: unfreeze fewer layers.
- different from pre-training: unfreeze more layers.

Benefits:

Frozen layers don't need gradient update hence reduced memory usage and faster training. Moreover, controlled adaptation prevents catastrophic forgetting.

TASK 4

Layer-wise learning rate rationale:

1. **Embedding layer:**

- Lowest Learning rate.
- Captures language statistics information.
- Changes affect higher layers.

2. **Base Transformer layers:**

- Gradually Increasing Learning rate.
- Lower layers captures basic language features, middle captures mixed features and upper layer captures task-specific features.
- Gradual increase preserves knowledge.

3. **Task specific head:**

- Highest Learning rate.
- Requires most adaptation and no pre-trained knowledge.

- **Independently set Learning rate for each task based on requirement.**

Benefits for training deep neural networks:
1. Preserving features in lower layer and also allowing faster adaptation in higher layers.
2. Helps get rid of vanishing gradient problem in the layers where gradient vanishes.
3. Better convergence and prevents catastrophic forgetting.

Benefits for Multi-task learning:
1. Multiple tasks can share basic features.
2. Higher layer can specialize for each task.