Lec 14 Combine Multiple Models

Good for getting slightly better acc

- Multiple architecture
- " initialisation
- " fine tuning

Model Ensembling

$$\begin{tikzcd}
 & M1 \arrow[r] & \hat{y}_1 \arrow[dr] & & \\
{} \arrow[ur] \arrow[dr] & & & \text{combine} \arrow[r] & \text{result} \\
 & M2 \arrow[r] & \hat{y}_2 \arrow[ur] & &
\end{tikzcd}$$

- Reduce bias
- Models may have seen different data

* Errors tend to be uncorrelated btwn models, ensembling can even them out

→ Can also ensemble across checkpoints

Doesn't need many data to train
Can be context dependent
↓

Ways to combine models:

1. Linear interpolation btwn model probs
   - ˪ interpolation coefficient can be constant or learnt
   - ˪ acts like logical OR btwn the two models
   - ˪ handles 0 prob
2. Log linear interpolation — on log probs then renormalise (softmax)
   - ˪ likewise, can be constant or learnt coefficients
   - ˪ acts like logical AND — high probs if all models high prob
   - ˪ allows negative coefficient — some model serve as negative evidence
     - ˪ eg. MT model + α(domain LM) − β(out of domain LM)
       LM + α(nontoxic LM) − β(toxic LM)

→ At test, drop out n times then combine
→ Bagging — resample dataset and train

Efficient multi model

cost $\propto$ amount of model

▹ Param averaging — average params of multiple models
- ˪ needs same archi & shapes (obviously)
- ˪ NNs have permutation invariance, so need same init.
- → Average together checkpoints (like last 5 of them)
- → Merge fine-tuned models

Model Soups paper
▹ Uniform averaging
▹ Greedy averaging — merge model if it improves
* Averaging perf correlates with ensembling perf usually

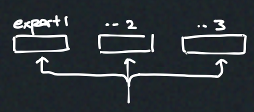
▹ Task vectors
- ˪ $v_{task} = \theta_{fine\text{-}tuned} - \theta_{original}$
- ˪ vector arithmetics to change task

▹ TIES to resolve conflict btwn multiple $\theta_{fine\text{-}tuned}$

→ Can use mergekit to do all these

▹ Ensemble Distillation
- ˪ make student model match the ensemble

▹ Sparse Mixture of Experts
- ˪ make use of $0 \cdot [\text{matrix}] = [0]$
- ˪ nvidia's cuSPARSE
- ▹ Sparsely Gated Mix of Experts Layer



← gate these experts, zero out some of them

Pipeline Systems

E2E can be hard:
- Data availability
- Interpretability

Cascading
$$X \rightarrow \boxed{M_1} \rightarrow \hat{Y}_1 \rightarrow \boxed{M_2} \rightarrow \hat{Y}$$

Stacking
$$\begin{tikzcd}
X \arrow[r] \arrow[rrr, bend right] & \boxed{M_1} \arrow[r] & \hat{Y}_1 \arrow[r] & \boxed{M_2} \arrow[r] & \hat{Y}
\end{tikzcd}$$

Iterative refinement
$$\begin{tikzcd}
X \arrow[r] & \boxed{M_1} \arrow[r] \arrow[loop below] & \hat{Y}
\end{tikzcd}$$