# 11711 HW1 Additional Improvements

**Keyu He**
Carnegie Mellon University
keyuhe@cmu.edu

## 1 Introduction

This short report documents the additional features I implemented for HW1: (1) attention masking of right-padding tokens during self-attention, and (2) flexible decoding controls (top-$k$, top-$p$ (nucleus), and beam search) for text generation.

## 2 Methods

### 2.1 Attention Mask on Padding Tokens

Right-padding is necessary to batch variable-length sequences, but it can introduce spurious attention to non-content positions. We compute a binary mask with *1* for real tokens and *0* for padding, broadcast it to shape $[\text{batch}, 1, 1, \text{seq\_len}]$, and apply it to the attention scores before softmax. Scores at padded key positions are set to $-\infty$, which zeroes their probability mass after softmax, ensuring padded tokens are never attended to. This reduces noise during training/evaluation, stabilizes gradients, and is especially helpful when there is substantial padding (longer sequences or high batch-size variability).

Usage: pass **`--use_pad_mask`** to enable this behavior. The flag is wired through data loading, the classifier, and the Llama attention implementation.

### 2.2 Decoding Controls for Generation

We expose several decoding strategies in `generate`:

- **Epsilon filtering** (existing method, default $\varepsilon = 0.05$): drop tokens with probability $< \varepsilon$. What I added is that if all are dropped (say, if used together with other decoding strategies), we fall back to the argmax token to keep a valid distribution.
- **Top-$k$** (`--top_k`): restrict sampling to the $k$ highest-probability tokens. Conceptually, this imposes a fixed-size candidate set; it sharpens choices but can be brittle if the true mass is spread across many tokens.
- **Top-$p$** (`--top_p`): nucleus sampling keeps the smallest set of tokens whose cumulative probability exceeds $p$. This yields a *dynamic* candidate set that adapts to the uncertainty of the distribution, preserving diversity when the distribution is flat and being selective when it is sharp.
- **Beam search** (`--beam_size`, `--beam_alpha`): maintains multiple partial hypotheses, expanding them by highest log-probability and selecting the best completed sequence. We use optional length normalization with exponent $\alpha$ to avoid short-sequence bias.

## 3 Experiments and Results

### 3.1 Classification with Attention Mask

We evaluate on SST and CFIMDB with and without the padding attention mask. Advanced outputs are saved as required: `sst-dev-advanced-output.txt`, `sst-test-advanced-output.txt`, `cfimdb-dev-advanced-output.txt`, `cfimdb-test-advanced-output.txt`.

| Dataset | Setting | Dev Acc | Test Acc |
|---|---|---|---|
| SST | Prompt (baseline) | **0.237** | **0.250** |
| SST | Prompt + Pad Mask (advanced) | 0.191 | 0.179 |
| CFIMDB | Prompt (baseline) | 0.490 | 0.109 |
| CFIMDB | Prompt + Pad Mask (advanced) | **0.498** | **0.756** |

Table 1: Accuracy with and without the attention pad mask.

### 3.2 Analysis of Results

**SST.** The prompt baseline slightly outperforms the masked variant on both dev and test. SST sentences are relatively short; there is limited right-padding, so masking offers little signal-to-noise benefit and can introduce minor distribution shifts

between training and evaluation. The small differences here are consistent with limited padding.

**CFIMDB.** We observe a large test improvement with masking (dev: 0.490 $\rightarrow$ 0.498; test: 0.109 $\rightarrow$ 0.756). CFIMDB reviews are longer with highly variable lengths, so batches contain substantial right-padding. Preventing attention to pads meaningfully improves the model's ability to focus on content tokens, which is reflected in the large test gain. The modest dev change suggests dev/test length distributions differ; masking helps most where padding is heavy.

### 3.3 Generation with Decoding Variants

We compare vanilla generation, top-$k$, top-$p$, and beam search using a fixed prompt. All generations are aggregated in `generated-sentences-advanced.txt`. Below are short excerpts illustrating qualitative differences (ellipses added for brevity):

**Vanilla (temp=0.0)**
*"... John Wick, is this day. He was playing with his toy car ... he had to be punished ... promised to never do it again."*

**Vanilla (temp=1.0)**
*"... John Wick, is the brave girl ... One day John was in his backyard ... 'I'm a hero' ... She was lost. He was ..."*

**Top-$k$ = 50 (temp=0.0)**
*"... John Wick, is this day. He was playing with his toy car ... promised to never do it again."*

**Top-$k$ = 50 (temp=1.0)**
*"... ran towards her ... 'Lets play!' ... She was a real hero ..."*

**Top-$p$ = 0.95 (temp=0.0)**
*"... John Wick, is this day. He was playing with his toy car ... promised to never do it again."*

**Top-$p$ = 0.95 (temp=1.0)**
*"... he had stolen a pizza! ... John returned the pizza and apologized ..."*

**Beam (size=5, $\alpha$=0.7, temp=0.0)**
*"... John Wick, is this day. He was playing with his toy car ... promised to never do it again."*

**Beam (size=5, $\alpha$=0.7, temp=1.0)**
*"... there too. He loves to play with his toys ... 'Let's play!' ..."*

## 4 Reproducibility

### 4.1 How to Run

- Run all experiments (includes classification and generation variants):
  `bash run_all_experiments.sh` (use `-no-gpu` on CPU-only machines)
- Advanced classification (pad mask) for SST/CFIMDB are invoked by the script via `--use_pad_mask`, saving to the required `*-advanced-output.txt` files.
- Generation variants are run and appended to `generated-sentences-advanced.txt`. To run a specific variant manually, e.g., top-$k$:
  `python run_llama.py --option generate --top_k 50`

## 5 Discussion

**What masking does.** It enforces a structural prior that self-attention should only consider real tokens. This reduces spurious correlations and improves stability when there is a lot of paddings.

**What beam search is.** A discrete search over sequence space that tracks $B$ best partial hypotheses and expands them by likelihood. Length normalization $\alpha$ mitigates favoring short completions. Beam search often improves grammaticality/consistency but can reduce diversity versus sampling.

**How $k$ and $p$ work conceptually.** Top-$k$ fixes the shortlist size (number of candidates); top-$p$ adapts the shortlist size to the model's uncertainty (sum of prob of the candidates, typically yielding more robust diversity across contexts. Epsilon acts as a light floor on probabilities.