Deriving and Using Embeddings in a Custom Logit Function

Compiled by D.Gueorguiev, 4/21/2025

Hidden states can be used to derive or refine embeddings. In pre-trained decoder-only LLMs (like GPT), embeddings are typically derived from the hidden state of the final layer. Hidden states can also be used to augment or enrich embeddings, especially in models like Transformers.

Here's a more detailed explanation:

1. Embeddings as an Initial Representation:

* In models like Transformers, embeddings are the initial numerical representations of input tokens (like words or characters).
* These embeddings are often learned during the model's training process, converting discrete inputs into continuous vector representations.
* Think of them as a "lookup table" where each token has a corresponding vector.

2. Hidden States as a Processed Representation:

* Hidden states, on the other hand, are the outputs of each layer in a neural network, including Transformers and RNNs.
* They represent the model's internal understanding of the input as it passes through the layers.
* Hidden states can be thought of as a "context-aware" representation, as they capture information about the input sequence.

3. How Hidden States Relate to Embeddings:

* **Deriving Embeddings:**

In pre-trained models, the final hidden state of the last layer is often used as the embedding for a token or sequence. This captures the model's learned representation of the input in its final form.

* **Enriching Embeddings:**

Hidden states can be used to augment or refine existing embeddings. For example, a token's embedding can be updated with the hidden state of its prediction, as described in a Reddit thread about enriching embeddings.

* **Capturing Context:**

Hidden states can also be used to capture context-aware representations of embeddings, which can be useful for tasks like language modeling or machine translation.

4. Examples:

* **BERT:**

In BERT, the last hidden state can be used as a context-aware representation of the input, which is more appropriate for sequence-based tasks than the unprocessed, context-invariant embeddings.

* **T5:**

In the T5 model, the hidden layer embeddings can be decoded for different decoding methods like beam search, [according to a Stack Overflow post](https://stackoverflow.com/questions/75305169/decoding-hidden-layer-embeddings-in-t5).

* [**LLMs**](https://www.google.com/search?cs=0&sca_esv=29e62d3773da02a2&sxsrf=AHTn8zp66JMQEmIHy3KZX7k3FE5EocdTnQ%3A1745266601075&q=LLMs&sa=X&ved=2ahUKEwjmtMvd-OmMAxWqK1kFHU-yErwQxccNegQIYRAB&mstk=AUtExfCfWRS4PBi7m9osyG5ppQ0KD3j1cTifyj8c0kmAFOiywjZTWm3bRaGsyKBcd2jGcSr_h-WmcsJDvbjV32x-H4cMXGpZt28EKmAuggbwhlih40MnCvmOnZZwr-X3gxvv_b8x7Rz44r45J3DJKkya38NQgHlS8wyU1R-dFsgBjfH_zkw&csui=3)**:**

In pre-trained decoder-only LLMs, embeddings are typically derived from the hidden state of the final layer, as described in [an arXiv paper](https://arxiv.org/html/2410.10814v2).

In summary: Embeddings are the initial, numerical representation of input tokens, while hidden states represent the model's processed understanding of the input at different layers. Hidden states can be used to derive, refine, or enrich embeddings, particularly in tasks where capturing context or fine-tuning embeddings is important.

References

[1] <https://discuss.huggingface.co/t/hidden-states-embedding-tensors/3549>