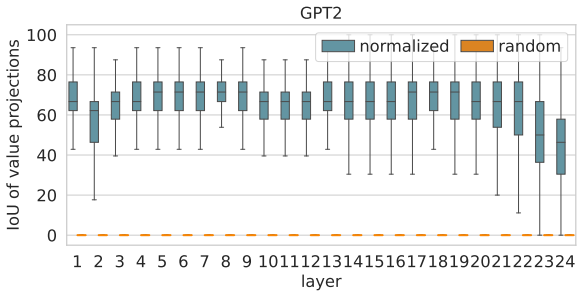

Figure 4: Similarity of projections to E, of GPT2 value vectors with and without layer normalization, and of value vectors and randomly-initialized vectors. We measure similarity of the top-30 tokens in each projection with IoU.

A Appendix

A.1 Value Vectors Projection Method

Our interpretation method of sub-updates is based on directly projecting value vectors to the embedding matrix, i.e. for a value $\mathbf{v}$ and embedding matrix E, we calculate $E\mathbf{v}$ (§4). However, in some LMs like GPT2, value vectors in each layer are added to the token representation followed by a layer normalization (LN) (Ba et al., 2016). This raises the question whether "reading" vectors that are normalized in the same manner as the representation would yield different concepts.

To test that, we compare the top-30 scoring tokens by $E\mathbf{v}_i^\ell$ and by $E \cdot \text{LayerNorm}(\mathbf{v}_i^\ell)$, for $i = 1, ..., d_m$ and $\ell = 1, ..., L$, using Intersection over Union (IoU). As a baseline, we also compare $E\mathbf{v}_i^\ell$ with random vectors, initialized from a normal distribution with the empirical mean and standard deviation of the value vectors. Fig. 4 shows that LN does not change the projection substantially, with an overlap of 64.5% of the top-30 tokens on average, suggesting that the same concepts are promoted in both cases. This is in contrast to random values, which produce a $\sim 0\%$ overlap on average.

A.2 Concepts Annotation

We analyze the concepts encoded in sub-updates, by projecting their corresponding value vectors to the embedding matrix and identifying repeating patterns in the top-scoring 30 tokens (§3). Pattern identification was performed by experts (NLP graduate students), following the instructions presented in Fig. 5. Please note these are the instructions provided for annotations of WIKILM, which uses word-level tokenization. Thus, the terms "words" and "tokens" are equivalent in this case.

For value vectors in WIKILM, which uses a word-level vocabulary with many uncommon words, we additionally attached a short description field for each token that provides context about the meaning of the word. For the description of a token w, we first try to extract the definition of w from Wordnet.[8] If w does not exist in Wordnet, as often happens for names of people and places, we then search for w in Wikipedia[9] and extract a short (possibly noisy) description if the query was successful. A complete annotation example Tab. 7.

A.3 Sub-Update Contribution in FFN Outputs

In this section, we justify our choice along the paper of looking at the top-10 dominant sub-updates. The contribution of a sub-update $m_i^\ell \mathbf{v}_i^\ell$ to the FFN output is:

$$\text{contrib}(m_i^\ell \mathbf{v}_i^\ell) := \frac{|m_i^\ell| ||\mathbf{v}_i^\ell||}{\sum_{j=1}^{d_m} |m_j^\ell| ||\mathbf{v}_j^\ell||},$$

namely, its relative weight compared to the overall sum of weights of all sub-updates. The overall contribution of the top-10 dominant sub-updates is computed by summing their contributions. Note that we take the absolute value of the coefficients $|m_i^\ell|$, since some activation functions (e.g. GeLU (Hendrycks and Gimpel, 2016) in GPT2), can result in negative values of m_i^ℓ.

Empirically, we observe that in some cases sub-updates with negative coefficients do appear as part of the 10 most dominant sub-updates in GPT2. We further attribute this to the success of GeLU in transformer models (Shazeer, 2020), as it increases the expressiveness of the model by allowing reversing the scores value vectors induce over the vocabulary.

Fig. 6 depicts the contribution of the top-10 dominant sub-updates per layer for WIKILM and GPT2, using 2000 random examples from the WIKITEXT-103 validation set. Clearly, for all the layers, the contribution of the dominant sub-updates exceeds the contribution of random sub-updates. Observe that, even though they cover only 0.24% of the value vectors, the contribution of dominant sub-updates is typically around 5%, and in some layers (e.g. layers 8-16 in WIKILM and layer 1 in GPT2) it reaches over 10% of the total

[8] We use the NLTK python package.

[9] Using the wptools package https://pypi.org/project/wptools/.

In this task, you are given a list of 30 words in English, and the goal is to identify repetitive patterns occurring in the words.
Patterns can be semantic (e.g. animals, 3-digit numbers, names of Indian actors, and time-related words) or syntactic (e.g. connectives, plurals, words starting with "dis-", and verbs in present progressive tense). You should only count patterns that occur in at least 4 words (i.e. if you notice a pattern that occurs only in 3 words, then please ignore it).

To complete the task, please do the following:

1. Give an ID to every identified pattern (1,2,...)
2. Assign a pattern ID to every word in the list, or -1/leave empty if no pattern applies to the word.
3. For every identified pattern specify whether the pattern is semantic or syntactic and (optional) write a short description of the pattern.

Please note that some of the words might be uncommon words that you are not familiar with. In such cases, you will need to do a quick search over the Web to understand the meaning of words.

Figure 5: Annotation instructions for the concepts identification task.

contribution. This demonstrates that analyzing the top-10 dominant sub-updates can shed light on the way predictions are built through the layers.

A.4 Toxic Language Suppression Details

The 10 manually selected value vectors were found by searching for non-toxic words, such as *"safe"* and *"peace"*, among the top-30 tokens in the vector projections to the vocabulary. We selected a small set of 10 value vectors whose top-scoring tokens were coherent and seemed to promote different kinds of non-toxic tokens. The list of manually picked vectors is provided in Tab. 8. Importantly, the search process of all vectors was a one-time effort that took < 5 minutes in total. We chose the value vectors in a greedy-manner, without additional attempts to optimize our choice.

To select 10 non-toxic value vectors based on an automatic toxicity metric, we used the Perspective API. Concretely, we concatenated the top-30 tokens by each value vector and graded the resulting text with the toxicity score produced by the API. Then, we sampled 10 random vectors with a toxicity score < 0.1 (a score of < 0.5 indicates a non-toxic text).

A.5 Early Exit Details

This section provides further details and analysis regarding our early exit method and the baselines we implemented.

Method Implementation. We consider 90% of the 10k examples for constructing T^ℓ and N^ℓ, and the remaining 10% examples are considered as the testing set. We used $k = 2e^2$ to cluster the top-10 dominant value vectors, but observed that other k values yielded similar results.

Baselines' Implementation. We train each binary classifier using 8k training examples, based on the standardized forms of each feature vector. We considered a hyperparameter sweep, using 8-fold cross-validation, with $l2$ or $l1$ regularization (lasso (Tibshirani, 1996) or ridge (Hoerl and Kennard, 1970)), regularization coefficients $C \in \{1e^{-3}, 1e^{-2}, 1e^{-1}, 1, 1e^1, 1e^2, 1e^3\}$, and took the best performing model for each layer. We also used a inversely proportional loss coefficient according to the class frequencies.

In order to achieve high accuracy, we further calibrate a threshold per classifier for reaching the maximal F_1 score for each layer. This calibration is done after training each classifier, over a set of 1000 validation examples.

Frequency of Saturation Events. We investigate the potential of performing early exit for WIKILM and GPT2. Tab. 9 and 10 depict the frequency of saturation events per layer, considering 10k examples from the WIKITEXT-103 validation set, for WIKILM and GPT2, respectively. In GPT2, 34.15% of the examples require the full computation using all the model layers, while for WIKILM, this holds for only 15.22% of the examples. Notably, early fixation events in GPT2 are less common than in WIKILM, possibly due to the larger number of layers the prediction construction is spread over. Hence, we use WIKILM for our experiments, as it has significantly higher compu-

patterns	word	description
1	front	the side that is forward or prominent
1	ahead	having the leading position or higher score in a contest
1	forward	the person who plays the position of forward in certain games, such as basketball, soccer, or hockey
1	preceded	be earlier in time; go back further
1	Before	earlier in time; previously
1	before	earlier in time; previously
1	rear	the back of a military formation or procession
1	fore	front part of a vessel or aircraft
2	Name	a language unit by which a person or thing is known
1	Past	the time that has elapsed
1	prior	the head of a religious order; in an abbey the prior is next below the abbot
1	anterior	a tooth situated at the front of the mouth
1	upperparts	standard terms for unambiguous description of relative placement of body parts
1	lead	an advantage held by a competitor in a race
1	backwards	at or to or toward the back or rear
1	aft	(nautical, aeronautical) situated at or toward the stern or tail
1	preceding	be earlier in time; go back further
1	upstream	in the direction against a stream's current
	hind	any of several mostly spotted fishes that resemble groupers
1	posterior	the fleshy part of the human body that you sit on
	Etymology	a history of a word
1	Pre	Wikimedia disambiguation page
	chin	the protruding part of the lower jaw
1	north	the region of the United States lying to the north of the Mason-Dixon line
1	east	the cardinal compass point that is at 90 degrees
2	surname	the name used to identify the members of a family (as distinguished from each member's given name)
1	Then	that time; that moment
2	name	a language unit by which a person or thing is known
1	northbound	moving toward the north
1	leading	thin strip of metal used to separate lines of type in printing

pattern id	description (optional)	semantic/syntactic
1	positions/ directions	semantic
2	naming	semantic

Table 7: An example annotation spreadsheet of the top-tokens by the value vector $\mathbf{u}^6_{1090}$ in WIKILM.

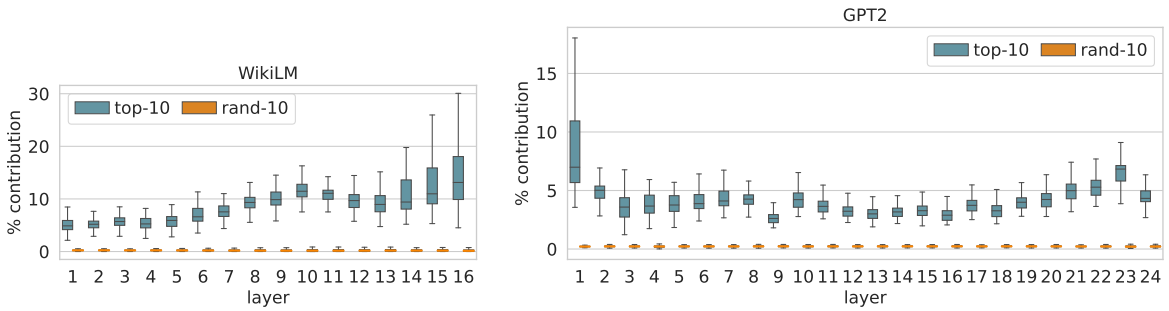

Figure 6: Relative contribution to the FFN output of the 10 most dominant and 10 random sub-updates in each layer, of WIKILM (left) and GPT2 (right).

tation saving potential, as well as more saturation events per layer.