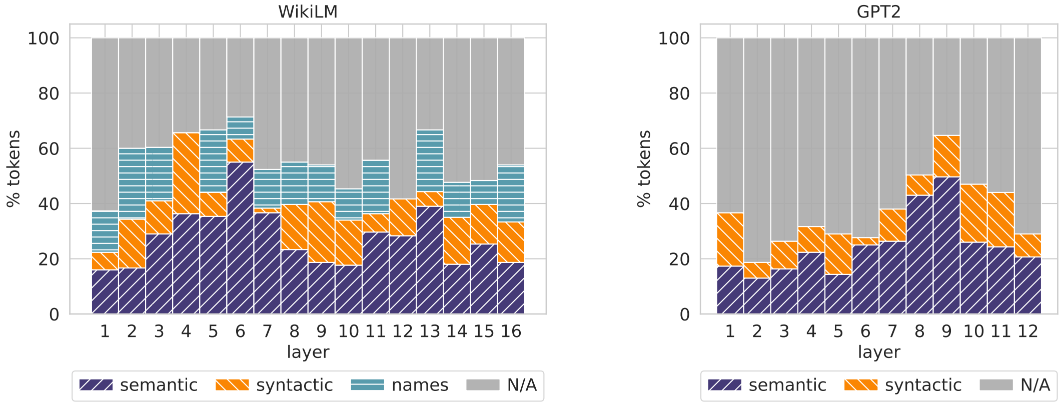

Figure 2: Portion of top-scoring tokens by value vectors in WIKILM and GPT2, that were associated with a semantic or syntactic concept, a name, or could not be matched to any concept ("N/A").

		Concept	Sub-update top-scoring tokens
GPT2	$\mathbf{v}^{3}_{1018}$	Measurement semantic	`kg, percent, spread, total, yards, pounds, hours`
	$\mathbf{v}^{8}_{1900}$	WH-relativizers syntactic	`which, whose, Which, whom, where, who, wherein`
	$\mathbf{v}^{11}_{2601}$	Food and drinks semantic	`drinks, coffee, tea, soda, burgers, bar, sushi`
WIKILM	$\mathbf{v}^{1}_{1}$	Pronouns syntactic	`Her, She, Their, her, she, They, their, they, His`
	$\mathbf{v}^{6}_{3025}$	Adverbs syntactic	`largely, rapidly, effectively, previously, normally`
	$\mathbf{v}^{13}_{3516}$	Groups of people semantic	`policymakers, geneticists, ancestries, Ohioans`

Table 1: Example value vectors in GPT2 and WIKILM promoting human-interpretable concepts.

corpus (Merity et al., 2017) with word-level tokenization ($|\mathcal{V}| = 267,744$), and GPT2 (Radford et al., 2019), a 12-layer LM trained on WEBTEXT (Radford et al., 2019) with sub-word tokenization ($|\mathcal{V}| = 50,257$). GPT2 uses the GeLU activation function (Hendrycks and Gimpel, 2016), while WIKILM uses ReLU, and in contrast to GPT2, WIKILM does not apply layer normalization after FFN updates. WIKILM defines $d = 1024, d_m = 4096$ and GPT2 defines $d = 768, d_m = 3072$, resulting in a total of $65k$ and $36k$ value vectors, respectively. For our experiments, we sample 10 random vectors per layer from each model, yielding a total of 160 and 120 vectors to analyze from WIKILM and GPT2, respectively.

4.1 Projection of Sub-Updates is Meaningful

Real vs. Random Sub-Updates. We validate our approach by comparing concepts in top-tokens of value vectors and 10 random vectors from a normal distribution with the empirical mean and standard deviation of the real vectors. We observe that a substantially higher portion of top-tokens were associated to a concept in value vectors compared to the random ones (Tab. 2): 55.1% vs. 22.7% in WIKILM, and 37% vs. 16% in GPT2. Also, in both models, the average number of concepts per vector was > 1 in the value vectors compared to ~ 0.5 in the random ones. Notably, no semantic nor syntactic concepts were identified in WIKILM's random vectors, and in GPT2, only 4% of the tokens were marked as semantic concepts in the random vectors versus 24.9% in the value vectors.

Updates vs. Sub-Updates. We justify the FFN output decomposition by analyzing concepts in the top-tokens of 10 random FFN outputs per layer (Tab. 2). In WIKILM (GPT2), 39.4% (46%) of the tokens were associated with concepts, but for 19.7% (34.2%) the concept was *"stopwords/punctuation"*. Also, we observe very few concepts ($< 4\%$) in the last two layers of WIKILM. We account this to extreme sub-updates that dominate the layer's output (§5.2). Excluding these concepts results in a considerably lower token coverage in projections of updates compared to those of sub-updates: 19.7% vs. 55.1% in WIKILM, and 11.8% vs. 36.7% in GPT2.

Overall, this shows that projecting sub-updates to the vocabulary provides a meaningful interface to the information they encode. Moreover, decomposing the FFN outputs is necessary for fine-grained interpretation of sub-updates.

	GPT2	WikiLM
FFN sub-updates	**36.7%**	**55.1%**
+ *stopwords concepts*	*37%*	*55.1%*
Random sub-updates	16%	22.7%
FFN updates	11.8%	19.7%
+ *stopwords concepts*	*46%*	*39.4%*

Table 2: Portion of top-scoring tokens associated with a concept, for FFN updates and sub-updates in WikiLM and GPT2, and for random vectors. For FFN updates/sub-updates, we show results with and without counting concepts marked as stopwords.

4.2 Sub-Update Projections are Interpretable

Fig. 2 shows a breakdown of the annotations across layers, for WikiLM and GPT2. In both models and across all layers, a substantial portion (40%-70% in WikiLM and 20%-65% in GPT2) of the top-tokens were associated with well-defined concepts, most of which were classified as *"semantic"*. Also, we observe that the top-tokens of a single value vector were associated with 1.5 (WikiLM) and 1.1 (GPT2) concepts on average, showing that *sub-updates across all layers encode a small-set of well-defined concepts*. Examples are in Tab. 1.

These findings expand on previous results by Geva et al. (2021), who observed that value vectors *in the upper layers* represent next-token distributions that follow specific patterns. Our results, which hold across *all the layers*, suggest that these vectors represent general concepts rather than prioritizing specific tokens.

Underestimation of Concept Frequency. In practice, we find that this task is hard for humans,[4] as it requires reasoning over a set of tokens without any context, while tokens often correspond to uncommon words, homonyms, or sub-words. Moreover, some patterns necessitate world knowledge (e.g. *"villages in Europe near rivers"*) or linguistic background (e.g. negative polarity items). This often leads to undetectable patterns, suggesting that the overall results are an underestimation of the true concept frequency. Providing additional context and token-related information are possible future directions for improving the annotation protocol.

Implication for Controlled Generation. If sub-updates indeed encode concepts, then we can not only interpret their contribution to the prediction, but also *intervene* in this process, by increasing the weights of value vectors that promote tendencies of our choice. We demonstrate this in §6.1.

$\mathbf{p}^{\ell}$: cow, cat, **dog**, goat, horse, bear
$\tilde{\mathbf{p}}^{\ell}$: **dog**, cat, goat, horse, cow, bear
Saturation: dog is promoted from rank 3 in $\mathbf{p}^{\ell}$ to rank 1 in $\tilde{\mathbf{p}}^{\ell}$, to be the top-candidate until the last layer.

$\mathbf{p}^{\ell}$: **cow**, cat, dog, goat, horse, bear
$\tilde{\mathbf{p}}^{\ell}$: dog, cat, goat, horse, **cow**, bear
Elimination: cow is eliminated from rank 1 in $\mathbf{p}^{\ell}$ to 5 in $\tilde{\mathbf{p}}^{\ell}$.

Table 3: Example saturation and elimination events, after a FFN update (reference tokens are in bold text).

5 FFN Updates Promote Tokens in the Output Distribution

We showed that sub-updates often encode interpretable concepts (§4), but how do these concepts construct the output distribution? In this section, we show that sub-updates systematically configure the prediction via promotion of candidate tokens.

5.1 Promoted Versus Eliminated Candidates

Every sub-update $m_i^{\ell}\mathbf{v}_i^{\ell}$ either increases, decreases, or does not change the probability of a token w, according to the score $\mathbf{e}_w \cdot m_i^{\ell}\mathbf{v}_i^{\ell}$ (§3). This suggests three mechanisms by which tokens are pushed to the top of the output distribution – *promotion*, where sub-updates increase the probability of favorable tokens, *elimination*, where sub-updates decrease candidate probabilities, or a *mixture* of both. To test what mechanism holds in practice, we analyze the scores sub-updates assign to top-candidate tokens by the representation. To simplify the analysis, we focus on changes induced by the 10 most dominant sub-updates in each layer, that is, the 10 sub-updates $m_i^{\ell}\mathbf{v}_i^{\ell}$ with the largest contribution to the representation, as measured by $|m_i^{\ell}| \cdot ||\mathbf{v}_i^{\ell}||$ (see details in App. A.3).

For the experiments, we use a random sample of 2000 examples from the validation set of WikiText-103,[5] which both WikiLM and GPT2 did not observe during training. As the experiments do not involve human annotations, we use a larger GPT2 model with $L = 24, d = 1024, d_m = 4096$.

We start by comparing the sub-updates' scores to a reference token in two types of events:

- *Saturation* (Tab. 3, up): The update $\mathbf{p}^{\ell} \rightarrow \tilde{\mathbf{p}}^{\ell}$ where the final token predicted by the model (i.e., $w = \text{argmax}(\mathbf{y})$) was promoted to be the top can-

[4] A sub-update annotation took 8.5 minutes on average.

[5] Data is segmented into sentences (Geva et al., 2021).

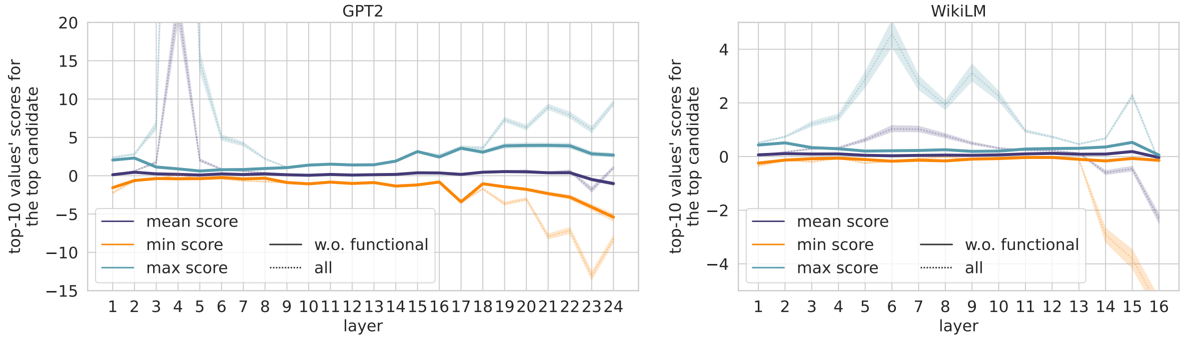

Figure 3: Mean, maximum and minimum scores assigned by the 10 most dominant sub-updates in each layer to the top-candidate token, in GPT2 (left) and WIKILM (right). Solid (dashed) lines exclude (include) functional value vector groups. The y-axis in both plots is cut for readability, as the max. (min.) scores reach 100 (-6).

Sub-updates	**Event**	**Max.**	**Mean**	**Min.**
WIKILM, dominant	*saturation* *elimination*	1.2 0.5	< 0.01 -0.01	-0.8 -0.5
WIKILM, random	*saturation* *elimination*	0.02 0.02	< 0.01 < 0.01	-0.02 -0.02
GPT2, dominant	*saturation* *elimination*	8.5 4.0	1.3 0.1	-4.9 -3.6
GPT2, random	*saturation* *elimination*	0.2 0.1	0.01 < 0.01	-0.2 -0.1

Table 4: Maximum, mean, and minimum scores of reference tokens in saturation and elimination events, by the 10 most dominant and 10 random sub-updates.

didate until the last layer. We analyze saturation events induced by the FFN before the last layer, covering 1184 and 1579 events in WIKILM and GPT2, respectively.

- *Elimination* (Tab. 3, bottom): The update $\mathbf{p}^\ell \to \tilde{\mathbf{p}}^\ell$ with the largest increase in the top candidate's rank, i.e. where the top candidate was dropped behind other candidates to have a rank > 1. Overall, our analysis covers 1909 (WIKILM) and 1996 (GPT2) elimination events.

We compute the mean, maximum, and minimum scores of the reference token by the 10 most dominant sub-updates in each event, and average over all the events. As a baseline, we compute the scores by 10 random sub-updates from the same layer.

Tab. 4 shows the results. In both models, tokens promoted to the top of the distribution receive higher maximum scores than tokens eliminated from the top position (1.2 → 0.5 in WIKILM and 8.5 → 4.0 in GPT2), indicating they are pushed strongly by a few dominant sub-updates. Moreover, tokens eliminated from the top of the distribution receive near-zero mean scores, by both dominant and random sub-updates, suggesting they are not being eliminated directly. In contrast to promoted tokens, where the maximum scores are substantially higher than the minimal scores (1.2 vs. -0.8 in WIKILM and 8.5 vs. -4.9 in GPT2), for eliminated tokens, the scores are similar in their magnitude (± 0.5 in WIKILM and 4.0 vs. -3.6 in GPT2). Last, scores by random sub-updates are dramatically lower in magnitude, showing that our choice of sub-updates is meaningful and that higher coefficients translate to greater influence on the output distribution.

This suggests that FFN updates work in a promotion mechanism, where top-candidate tokens are those being pushed by dominant sub-updates.

5.2 Sub-Updates Across Layers

To analyze the FFN operation in different layers, we break down the top-candidate scores per layer. Formally, let $w^\ell = \text{argmax}(\mathbf{p}^\ell)$ be the top candidate at layer ℓ (before the FFN update) for a given input, we extract the scores $\mathbf{e}_{w^\ell} \cdot m_i^\ell \mathbf{v}_i^\ell$ by the 10 most dominant sub-updates and compute the mean, minimum and maximum scores over that set.

Fig. 3 shows that, in both models, until the last few layers (23-24 in GPT2 and 14-16 in WIKILM), maximum and minimum scores are distributed around non-negative mean scores, with prominent peaks in maximum scores (layers 3-5 in GPT2 and layers 4-11 in WIKILM). This suggests that the token promotion mechanism generally holds across layers. However, scores diverge in the last layers of both models, with strong negative minimum scores, indicating that the probability of the top-candidate is pushed down by dominant sub-updates. We next show that these large deviations in positive and negative scores (Fig. 3, dashed lines) result from the operation of small sets of functional value vectors.