Conceptizer: Next-Token Prediction Is Not All You Need

Haoran Zheng
The University of Chicago
haoranzheng@uchicago.edu

Chen Shani
Stanford University
Tel Aviv University
cshani@tauex.tau.ac.il

arXiv:2610.33027v1 [cs.CL] 11 Oct 2026

Abstract

Next-token prediction (NTP) rewards **only** the token that occurs next, penalizing every alternative. But natural language often offers many ways to express the same meaning. We introduce Conceptizer, a training framework that extends next-token prediction with a reward for *meaning*. Conceptizer uses a fixed, model-agnostic vocabulary of ∼9K concepts, built from contextual synonyms, as units of meaning alongside tokens as units of text. It maps each next-token distribution to a concept distribution and shares the reward for the target meaning across its possible wordings. This provides a concept-level training signal without adding model parameters. Applied to 3 model families at 1B-8B parameters, Conceptizer achieves the lowest held-out NTP perplexity at concept-target positions and assigns the highest probability to contextual synonyms, against all 3 baselines. It also improves downstream performance on knowledge and sentiment tasks while remaining comparable on commonsense. These results show that language models can be directly optimized to predict meaning, not just the next token.

1 Introduction

Next-token prediction (NTP) directs a language model (LM) toward a single target at each position: the observed next token in the corpus. After "...practice the following sun", a base model gives *protection* and *safety* 32% probability each, yet NTP rewards only the observed word, *safety*, and pushes *protection* down as it would a wrong word.

The units that NTP scores are built from surface statistics, not meaning: standard tokenizers partition text based on statistical regularities rather than semantic relationships. We introduce *Conceptizer*,[1] a training framework that adds a fixed vocabulary of units of meaning alongside each model's existing tokenizer. In this vocabulary, *safety*, *protection* and *security* can share concept #1450, allowing the training objective to reward different wordings of the same contextual meaning.

Offline, we construct a vocabulary of 9,061 concepts (Fig. 1). Unlike prior work, this vocabulary is built once from *contextual synonyms* (lexical substitutes; McCarthy and Navigli, 2007) and shared across models. During training, a *concept loss* turns each next-token distribution into a distribution over concepts and pulls it toward a target built from the next token's contextual synonyms (Fig. 2).

Inference is unchanged from standard NTP: the final softmax and token vocabulary remain the same. The only difference is that during training, **Conceptizer rewards all tokens that express the same underlying concept**.

We post-train Llama 3, Qwen2.5 and Falcon3 models (Meta, 2024a; Grattafiori et al., 2024; Qwen Team, 2024; Falcon-LLM Team, 2024) at 3 scales (1B to 8B). We then compare Conceptizer with 3 baselines: the base model, next-token prediction post-training (NTP post-training) and a Randomized synonyms control. At held-out concept-target positions, Conceptizer assigns higher NTP probability to the observed word and its contextual synonyms than all 3 baselines (Fig. 4; Appendix F). On downstream evaluations, Conceptizer outperforms the baselines in 26 of 27 comparisons on knowledge and sentiment tasks, while remaining competitive on commonsense (Fig. 5).

2 Related Work

Concept-aware language models. Shani et al. (2023) group a model's predictions into concepts, Iyer et al. (2026) post-train a model to predict a noun's synonyms and hypernyms, and Zhang et al. (2026) extend this to all content words, pooling the probability of contextual synonyms drawn from the

[1] Code, the concept vocabulary and the training data: https://github.com/HaoranZhengRaul/conceptizer-code.

model's own predictions, one occurrence at a time. Conceptizer instead builds a fixed concept vocabulary shared by every word, occurrence and model, and its concept loss contains the losses of Iyer et al. and Zhang et al. as special cases (Appendix B).

Tokenization. Token vocabularies are built from surface statistics, not meaning: BPE merges frequent pairs of symbols (Gage, 1994; Sennrich et al., 2016), WordPiece and unigram tokenization choose units by likelihood (Schuster and Nakajima, 2012; Wu et al., 2016; Kudo, 2018), and SentencePiece and byte-level BPE apply them to raw text and bytes (Kudo and Richardson, 2018; Radford et al., 2019). So no token vocabulary connects *safety*, *protection* and *security*. Conceptizer keeps each model's tokenizer and adds concepts alongside it.

Words grouped by meaning. Surface forms of one answer compete for probability (Holtzman et al., 2021), and semantic entropy pools the answers that share a meaning to estimate a model's uncertainty (Kuhn et al., 2023). Grouping words by meaning has a long lineage: class-based language models factor a word's probability through its class (Brown et al., 1992; Goodman, 2001); SenseBERT also predicts one of WordNet's 45 supersenses through a learned head, pooling the probability of the supersenses WordNet allows for the word, independent of its context (Levine et al., 2020; Miller, 1995); LIBERT adds the classification of word relations such as synonymy to BERT's pretraining (Lauscher et al., 2020); and Backpack models learn sense vectors per word inside the network (Hewitt et al., 2023). Conceptizer finds each word's senses from its lexical substitutes (McCarthy and Navigli, 2007) across sentences, as in substitute-based word sense induction (Amrami and Goldberg, 2018; Eyal et al., 2022), and merges the senses of different words into concepts. A model then reaches these concepts through concept memberships, without new parameters, and each target comes from its occurrence's own contextual synonyms.

Soft targets. Softening the one-hot target also has a long history: label smoothing mixes it with a uniform distribution (Szegedy et al., 2016), semantic variants spread the credit over related tokens or sequences (Elbayad et al., 2018; Lukasik et al., 2020), D2GPo adds a KL term toward a Gaussian prior over tokens ordered by embedding similarity (Li et al., 2020), and AdaLabel learns a context-dependent token target (Wang et al., 2021). Distillation trains toward a teacher's distribution (Hinton et al., 2015; Kim and Rush, 2016), as our retention loss does toward the frozen base model, and selective language modeling keeps the loss only on the tokens a reference model marks as useful (Lin et al., 2024). These losses score each token or sequence on its own, while the concept loss scores the total probability of each concept. Partial-label learning also scores a set of candidate labels together, but only one of them is correct (Cour et al., 2011; Cid-Sueiro, 2012).

Concepts as units of generation. Other work predicts or reasons with "concepts" and other latent units, each under its own definition of a concept. Large Concept Models take a sentence's embedding as a concept and predict the next one (LCM team et al., 2024). CoCoMix takes features of a sparse autoencoder as concepts, predicts them and mixes them into the model's hidden states (Tack et al., 2026). ConceptLM takes a learned code for each chunk of 4 tokens as a concept and predicts it to guide the next tokens (Liu et al., 2026). Soft Thinking, Coconut and Soft Tokens, Hard Truths reason by feeding back a continuous vector, a probability-weighted mixture of token embeddings or the model's last hidden state, in place of the token sampled at each step (Zhang et al., 2025; Hao et al., 2025; Butt et al., 2026). These "concepts" are learned or continuous representations, not lists of words. For Conceptizer, a concept is an explicit set of words that can replace one another in a context without changing its meaning, such as #1450's *safety*, *protection* and *security*.

3 Data Preparation: Concept Vocabulary and Targets

Data. We use FineWeb-Edu, a collection of English educational web documents (Penedo et al., 2024). We tag nouns, verbs and adjectives with spaCy (Honnibal et al., 2020) and refer to these as *content words*. For each word, we collect up to 25 occurrences with the same part of speech (POS), grouping all morphological forms together: *documented* and *documenting*, for example, are both treated as (document; verb). We discard (word; POS) combinations with fewer than 3 occurrences. This yields 2.33 million occurrences of 172,925 words (13.5 per word on average). The concept vocabulary $\mathcal{C}$ is constructed from 88,367 of these occurrences. Let w_0 denote the *observed word*, such as *safety* after "…practice the following sun"

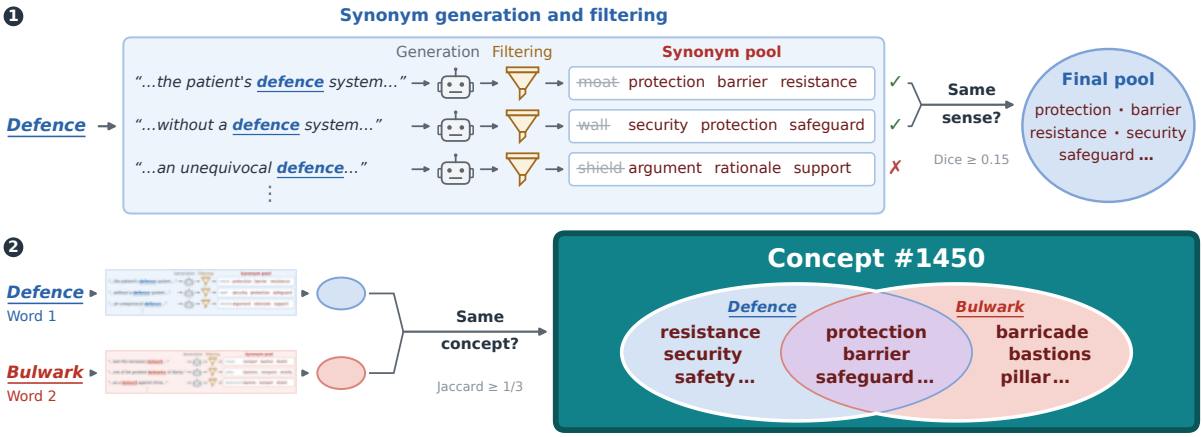


Figure 1: **Building Conceptizer's fixed, model-agnostic concept vocabulary.** For up to 25 FineWeb-Edu occurrences of each content word, LLMs and WordNet propose synonym candidates, and an LLM judge keeps those that fit the sentence. Dice similarity groups one word's synonym pools into senses, and Jaccard similarity merges the senses of different words into 9,061 concepts (Section 3; Appendices C and E).

(our running example).

Synonym candidate generation. For each collected occurrence, 3 sources propose *synonym candidates*. Llama-3.3-70B-Instruct (Meta, 2024b) proposes contextual replacements for the sense in the sentence, while Qwen2.5-72B-Instruct (Qwen Team, 2024) and WordNet (Miller, 1995) see only the word and its POS and cover all its senses. This casts a wide net, which the LLM judge narrows.

LLM judge. All candidates are input into an *LLM judge*, a frozen gpt-oss-120B (OpenAI, 2025). It is instructed to keep the candidates that can replace the observed word w_0 without changing the sentence's meaning or grammatical form. The judge's answers vary between runs (temperature 0.7), so one run can miss a borderline synonym. We run it 3 times and keep any candidate one run keeps (details in Appendix C). In the same answer, the judge can add up to 2 contextual synonyms to avoid obvious misses by the candidate sources. These kept words form the occurrence's *synonym pool* ("concept set" in Zhang et al., 2026). Of 2.33 million occurrences, 1.84 million have a synonym pool; the remainder are mostly names, codes and specialized terms such as *insulin*, for which no suitable candidate replacement is found. For simplicity, synonyms that the trained model's tokenizer splits into several tokens are dropped, and the rest, at most 10 (the cap of Zhang et al., 2026), are labeled $w_1, \ldots, w_m$. For example, for (safety; noun) after "...practice the following sun", the judge keeps 3 of 43 candidates: $\{w_1 = \textit{protection}, w_2 = \textit{prevention}, w_3 = \textit{care}\}$ (it cut *security*, among others).

Judge weights. Each synonym candidate is then weighted by how often the LLM judge keeps it when it is proposed, across all collected occurrences of w_0 (up to 25). The 3 judge runs of any single occurrence give each candidate one binary verdict, not a fraction such as 2/3. If a synonym w_j is proposed in n_j of these occurrences and kept in k_j, its *judge weight* r_j is the Laplace-smoothed rate

$$r_j = \frac{k_j + 1}{n_j + 2}, \tag{1}$$

and w_0 receives $r_0 = 1$. Over the 25 occurrences of *safety* (Table 3 in Appendix C), *protection* is proposed and kept in all 25 ($r_1 = 26/27 = 0.96$), *prevention* is proposed in 6, once by the judge, and kept in 4 ($r_2 = 5/8 = 0.63$), and *care* is kept in 2 of 25 ($r_3 = 0.11$). Smoothing keeps thin evidence from looking certain: *carefulness*, kept the one time it was proposed, gets a judge weight $r_j = 2/3$ instead of 1. A synonym's judge weight r_j is the same for all collected occurrences of w_0, but each occurrence uses it only for its own synonym pool: *security*, kept in 22 of 25, goes unused for "...practice the following sun", where the judge cut it.

The concept vocabulary. We build the *concept vocabulary* $\mathcal{C}$ from the occurrences of 5,939 frequent content words (3.8% of all collected; Fig. 1; Appendix C). First, to distinguish their senses, we group each word's synonym pools into *senses* by Dice similarity ($2|A \cap B|/(|A| + |B|) \geq 0.15$ for pools A and B; Dice, 1945). A sense's *final pool* is the word itself plus the union of its qualified syn-

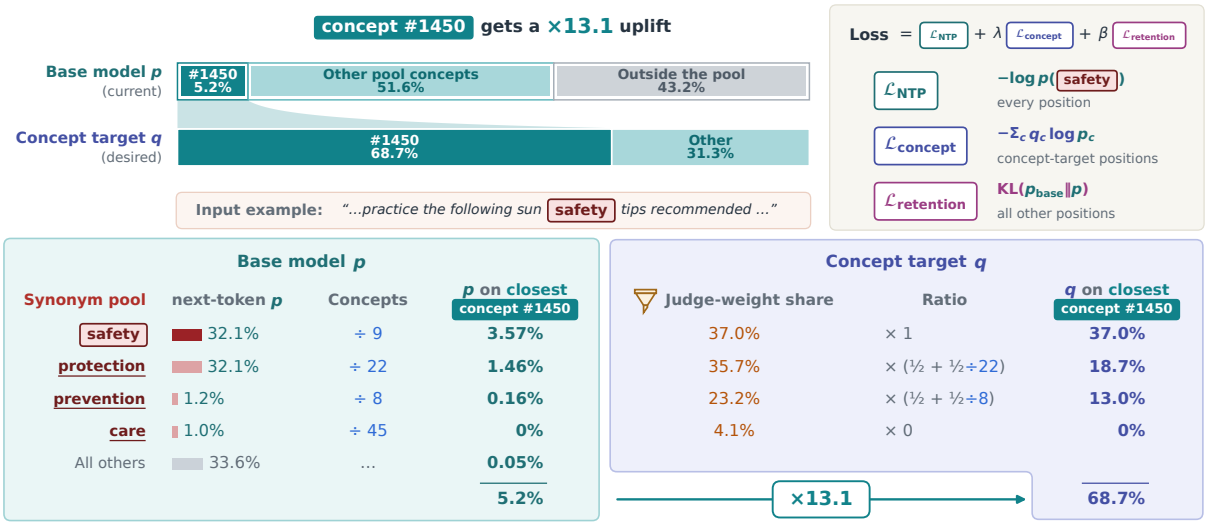


Figure 2: **Concept training with Conceptizer.** Concepts: each token's probability is split evenly over the concepts it has a membership in. Ratio κ_j: the fraction of each word's judge-weight share ρ_j sent to the closest concept $c^\star$, #1450. Uplift: $q \div p$ on #1450. *Care* is not in #1450, so its p and q on #1450 are 0; all its concepts are other pool concepts. Other pool concepts: all other concepts that *protection*, *prevention* and *care* belong to; outside the pool: the remaining concepts, and tokens in no concept, such as *-related*.

onym pools. Second, to create each *concept*, we merge senses of different words by Jaccard similarity ($|A \cap B|/|A \cup B| \geq 1/3$ for final pools A and B, with at least 2 shared words; Jaccard, 1912). A concept's *member words* are the union of these final pools. The resulting concept vocabulary $\mathcal{C} = \{c_0, \ldots, c_{9060}\}$ (c_n is written #n) contains 49,387 member words, far exceeding the original 5,939, and is model-agnostic: the same 9,061 concepts serve all 9 models we post-train.

Concept memberships. Concept memberships connect each model's output tokens to our shared concept vocabulary $\mathcal{C}$. The vocabulary is the same across models, but their tokenizers differ, so the same word may correspond to different tokens in different models. We therefore construct the mapping separately for each tokenizer.

A token belongs to a concept if the corresponding word appears in that concept. We also include a token when it is the first token of a longer word in that concept (a *first-token membership*). A token can belong to multiple concepts. For example, the token for *safety* belongs to 9 concepts, among them #1450 and #6387, which lists *safetylines* but not *safety*. In total, Falcon3 has 23,992 words represented as single tokens that participate in at least one concept, compared with 14,836 for Llama and 14,829 for Qwen2.5.

Let $\mathcal{V}$ be the model's output token vocabulary and $\mathcal{H}(v)$ be the *concept memberships* of token v, the set of concepts it is associated with. A token in no concept has $\mathcal{H}(v) = \{\perp\}$, the *no-concept outcome*, giving $\mathcal{C}^+ = \mathcal{C} \cup \{\perp\}$. When a token belongs to multiple concepts, its *membership weights* ω_{vc} divide its contribution equally among them:

$$\omega_{vc} = \frac{\mathbf{1}[c \in \mathcal{H}(v)]}{|\mathcal{H}(v)|}, \qquad v \in \mathcal{V},\ c \in \mathcal{C}^+. \quad (2)$$

The concept target. At each selected *concept-target position* i, Conceptizer adds a *concept target* $\mathbf{q}_i$, a target distribution over $\mathcal{C}^+$ (the concept vocabulary $\mathcal{C}$ and $\perp$), to the standard next-token target. These positions contain content words that are represented as a single whole-word token and have an associated concept (Appendix C).

We construct $\mathbf{q}_i$ from the observed word w_0 and its judge-weighted contextual synonyms $w_1, \ldots, w_m$. First, we identify the *closest concept* $c^\star$: among the concepts containing w_0, it is the one that contains the most contextual synonyms $w_1, \ldots, w_m$. Each word w_j contributes its normalized judge weight (*judge-weight share*) $\rho_j = r_j / \sum_{l=0}^{m} r_l$, where r_j is its judge weight (Equation 1).

We then distribute each word's share ρ_j across its associated concepts $\mathcal{H}(w_j)$. The observed word w_0 assigns all of its share ρ_0 to $c^\star$ ($\sigma_0 = 1$). A synonym whose token also belongs to $c^\star$ assigns half of its share ρ_j to $c^\star$ ($\sigma_j = 1/2$) and distributes the remaining half equally across all its concept memberships $\mathcal{H}(w_j)$, $c^\star$ included. A synonym whose

token does not belong to $c^\star$ ($\sigma_j = 0$) distributes its entire share ρ_j equally across its concept memberships $\mathcal{H}(w_j)$. Thus, the target combines direct evidence for the closest concept $c^\star$ with the other meanings associated with each synonym:

$$\mathbf{q}_i = \sum_{j=0}^{m} \rho_j \left[\sigma_j \, \mathbf{e}_{c^\star} + (1 - \sigma_j) \, \boldsymbol{\omega}_{w_j}\right], \quad (3)$$

where $\mathbf{e}_{c^\star}$ is $c^\star$'s unit vector, $\boldsymbol{\omega}_{w_j} = (\omega_{w_j c})_{c \in \mathcal{C}^+}$ gives the membership weights of w_j's token (Equation 2), and σ_j is the part of ρ_j sent directly to $c^\star$.

The resulting $\mathbf{q}_i$ is a probability distribution over $\mathcal{C}^+$ ($\mathcal{C}$ and $\perp$), sums to 1 and is fixed by the training data rather than the model's predictions. Depending on the tokenizer, 298,227 to 475,558 training positions qualify for a concept target. The concept loss is applied at these positions.

4 Training Conceptizer

We post-train a base model with a LoRA adapter (Hu et al., 2022) on FineWeb-Edu text, using 3 losses: NTP $\mathcal{L}_{\text{NTP}}$ at every position, a concept loss $\mathcal{L}_{\text{concept}}$ at concept-target positions $\mathcal{S}$ and a *retention loss* $\mathcal{L}_{\text{retention}}$ at all other positions $\mathcal{U}$, which keeps the model's next-token distribution there close to the base model's.

Model concept distribution. At prediction position i, the logits $\mathbf{z}_i$ give the next-token distribution $\mathbf{p}_{\text{w},i} = \text{softmax}(\mathbf{z}_i)$ over the output token vocabulary $\mathcal{V}$. The membership weights ω_{vc} of Equation 2 split each token's probability evenly over its concepts $\mathcal{H}(v)$, and summing per concept gives the *model concept distribution* $\mathbf{p}_{\text{c},i}$ that sums to 1:

$$p_{\text{c},i}(c) = \sum_{v \in \mathcal{V}} \omega_{vc} \, p_{\text{w},i}(v), \qquad c \in \mathcal{C}^+. \quad (4)$$

Losses and objective. In one optimizer step, let $\mathcal{I}$ be the positions that predict a next token, g_i the observed next token at position i ($g_i = w_0$ at a concept-target position), $\mathcal{S} \subseteq \mathcal{I}$ the concept-target positions, $\mathcal{U} = \mathcal{I} \setminus \mathcal{S}$ the other positions, and $\mathbf{p}^{\text{base}}_{\text{w},i}$ the frozen base model's next-token distribution. NTP, $\mathcal{L}_{\text{NTP}}$, is the cross-entropy of the observed next token g_i, the concept loss $\mathcal{L}_{\text{concept}}$ the cross-entropy between the concept target $\mathbf{q}_i$ (Equation 3) and the model concept distribution $\mathbf{p}_{\text{c},i}$ (Equation 4), and the retention loss $\mathcal{L}_{\text{retention}}$ the KL divergence between the base model's and the model's next-token distributions, $\mathbf{p}^{\text{base}}_{\text{w},i}$ and $\mathbf{p}_{\text{w},i}$:

$$\mathcal{L}_{\text{NTP}} = -\frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \log p_{\text{w},i}(g_i), \quad (5)$$

$$\mathcal{L}_{\text{concept}} = -\frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \sum_{c \in \mathcal{C}^+} q_i(c) \log p_{\text{c},i}(c), \quad (6)$$

$$\mathcal{L}_{\text{retention}} = \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} D_{\text{KL}}\left(\mathbf{p}^{\text{base}}_{\text{w},i} \,\|\, \mathbf{p}_{\text{w},i}\right), \quad (7)$$

and the objective $\mathcal{L}$ is

$$\boxed{\mathcal{L} = \mathcal{L}_{\text{NTP}} + \lambda \, \mathcal{L}_{\text{concept}} + \beta \, \mathcal{L}_{\text{retention}}}, \quad (8)$$

with loss weights $\lambda = 0.6$ and $\beta = 0.5$, held constant across models and methods (hyperparameter sweeps in Appendix E). Each loss is averaged over its own positions in the optimizer step: each concept target $\mathbf{q}_i$ enters the objective $\mathcal{L}$ with weight $\lambda/|\mathcal{S}|$ and each NTP term with $1/|\mathcal{I}|$.

How the concept loss acts. At the running example (Fig. 2), the judge weights $r_0, \ldots, r_3$ of *safety*, *protection*, *prevention* and *care*, 1, 0.96, 0.63 and 0.11, sum to 2.70, giving judge-weight shares $\rho_0, \ldots, \rho_3$ of 37%, 36%, 23% and 4%. The concept target puts $q_i(c_{1450}) = 68.7\%$ on the closest concept $c^\star$, #1450: all of *safety*'s share, half of *protection*'s and *prevention*'s, and 1/22 and 1/8 of their other halves. The remaining 31.3% lies on the 68 other concepts of *protection*, *prevention* and *care*. At the first optimizer step, the model still equals the base model, because its LoRA adapter starts at zero. The model concept distribution then gives #1450 only $p_{\text{c},i}(c_{1450}) = 5.23\%$, mostly from *safety*'s and *protection*'s 32.1% each, split over 9 and 22 concepts. At that step, the gradient of the concept loss $\mathcal{L}_{\text{concept}}$ acts in 3 ways.

First, like any cross-entropy (Equation 6), the concept loss $\mathcal{L}_{\text{concept}}$ pulls each concept c by its **uplift**, the concept target divided by the model concept distribution, $q_i(c)/p_{\text{c},i}(c)$. For #1450 the uplift is $68.7/5.23 = 13.1$: the target $\mathbf{q}_i$ asks 13.1 times what the model concept distribution $\mathbf{p}_{\text{c},i}$ gives. A concept is *under-served* when the model concept distribution $\mathbf{p}_{\text{c},i}$ gives it less than the target $\mathbf{q}_i$ asks (uplift above 1) and *over-served* when it gives more (uplift below 1; Appendix A). A concept outside the target $\mathbf{q}_i$ has an uplift of 0.

Second, the concept memberships $\mathcal{H}(v)$ pass each concept's uplift down to its tokens. A token v takes from each of its concepts an *uplift share*, the concept's uplift divided by the token's number of

concepts $|\mathcal{H}(v)|$, because the token gives each of its concepts that fraction of its probability in the model concept distribution $\mathbf{p}_{c,i}$ (Equation 4). The sum of the token v's uplift shares over all its concepts $\mathcal{H}(v)$ is its **pull**, $\sum_c \omega_{vc}\, q_i(c)/p_{c,i}(c)$. *Safety*'s token, in 9 concepts, takes an uplift share of $13.1/9 = 1.46$ from #1450, and its pull is 1.58. *Protection*'s token, in 22 concepts, takes only $13.1/22 = 0.60$ from #1450, and its pull is 1.06. A token with few concepts is pulled strongly, and one with many barely. The token's **push**, the change the loss asks for in its logit $z_i(v)$, is $p_{w,i}(v)$ (pull -1), where the -1 comes from the softmax: the logit $z_i(v)$ rises when v's pull is above 1 and falls when it is below 1. *Safety*'s push is $0.321 \times (1.58 - 1) = +0.19$, and *protection*'s $0.321 \times (1.06 - 1) = +0.02$.

Third, the concept loss $\mathcal{L}_{\text{concept}}$ rewards or penalizes the words of a concept together. Every token of #1450 takes an uplift share from it, not only those in this occurrence's synonym pool. *Security*, cut by the judge in this occurrence, has a token in 9 concepts, #1450 among them, so it takes the uplift share of 1.46 and its logit rises. *Security*'s push is only $+0.0004$, against *safety*'s $+0.19$, because the base model gives it only 0.07%. Next-token prediction and the concept losses of Iyer et al. (2026) and Zhang et al. (2026) are special cases of the Conceptizer concept loss $\mathcal{L}_{\text{concept}}$ (Appendix B).

Pulls and pushes change at every step as the model's probabilities move, so the first step does not predict the end of training. Read at the same training position after 50,000 steps (Fig. 3; details in Appendix D), Llama 1B's Conceptizer adapter gives *protection* 42.11% against NTP post-training's 38.55%, *care* 1.85% against 1.03% and *safety* 25.54% against 34.02%. Together, *safety* and its contextual synonyms w_1, w_2, w_3 hold 71.35% of the next-token probability $\mathbf{p}_{w,i}$ after training.

5 Implementation Setup

Models. We post-train 9 models: Llama-3.2-1B/3B-Instruct (Meta, 2024a) and Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Qwen2.5-1.5B/3B/7B-Instruct (Qwen Team, 2024), and Falcon3-1B/3B/7B-Instruct (Falcon-LLM Team, 2024).

Methods. The **Base model** is the released checkpoint, **NTP post-training** trains it with NTP and retention losses, $\mathcal{L}_{\text{NTP}}$ and $\mathcal{L}_{\text{retention}}$, and **Conceptizer** adds the concept loss $\mathcal{L}_{\text{concept}}$. **Randomized synonyms** is post-trained like Conceptizer,

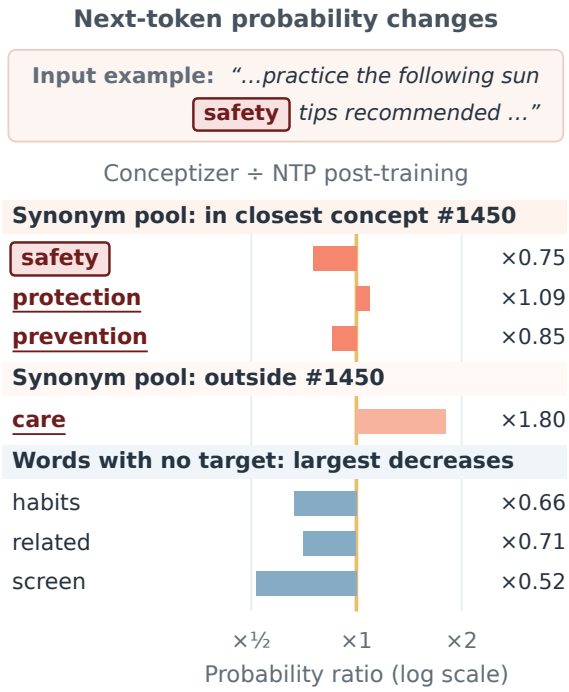


Figure 3: **Conceptizer raises the next-token probability of *protection* and *care* and lowers that of *safety* and *prevention*.** After 50,000 steps (Llama 1B, seed 0).

but each concept-target position gets the contextual synonyms $w_1, \ldots, w_m$ and judge weights r_j of another target with as many synonyms. The position's closest concept $c^\star$ is also redrawn uniformly from the observed token's concepts $\mathcal{H}(w_0)$, so the observed word w_0's share ρ_0 goes to a random one.

Training. All post-trained methods share the same training text, order and recipe: 50,000 optimizer steps of a LoRA adapter on FineWeb-Edu, with seeds 0, 37 and 42 (Appendix D). Concept-target positions are 0.15 to 0.24% of all positions.

6 Experiments and Results

6.1 Prediction on Held-Out Text

We hold out 10% of the documents from training (Appendix C) and split them between development and final evaluation. NTP perplexity is measured on the final part, 15,909 to 15,920 documents and 9.74 to 9.81 million predicted tokens per family, at 3 sets of positions: concept-target positions (5,154 for Llama and Qwen2.5, 10,028 for Falcon3), *content-word positions* (each occurrence's first token) and all tokens (*global perplexity*).

Results. At held-out concept-target positions, Conceptizer gives the observed word w_0 the most probability against the 3 baselines in every

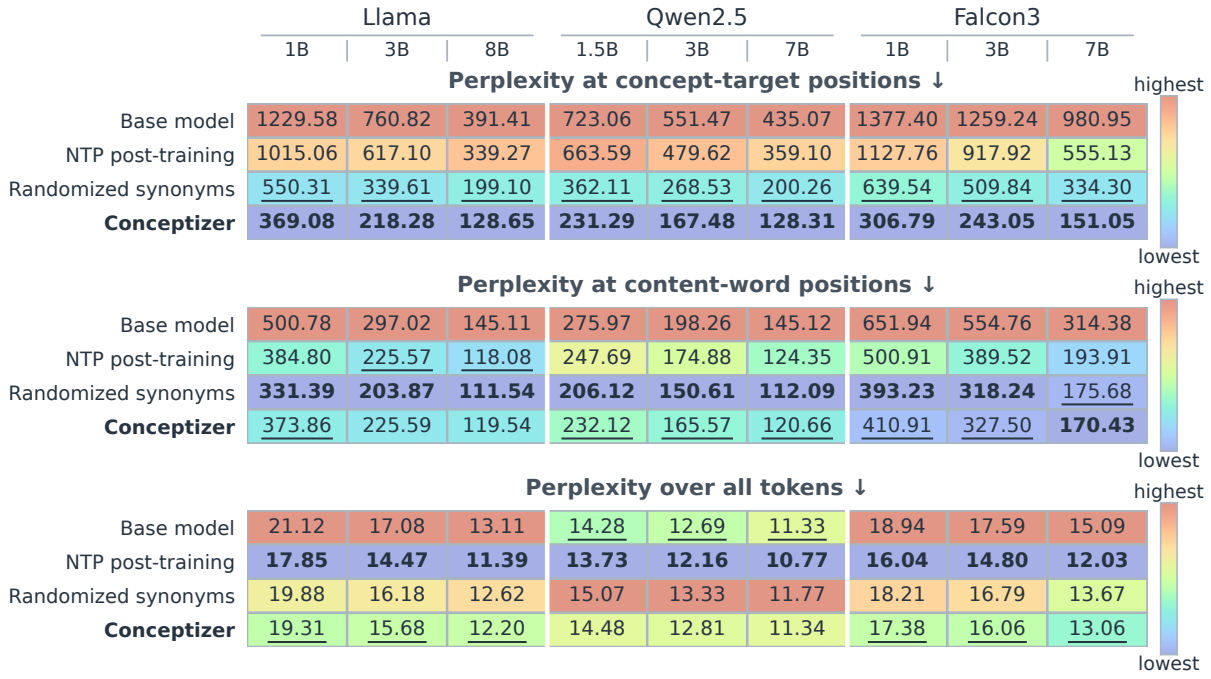


Figure 4: **Conceptizer has the lowest perplexity against the 3 baselines at concept-target positions in every model.** NTP post-training has the lowest global perplexity in every model, and Randomized synonyms the lowest on content words in 8 of 9. Held-out means of training seeds 0, 37 and 42. Colors scale per model and panel.

model (Fig. 4; Table 16): in geometric mean, 2.6 to 3.8 times NTP post-training's, 1.5 to 2.2 times Randomized synonyms' and 3.0 to 6.5 times the base model's. For Llama 1B, the observed word w_0's geometric-mean probability over the 5,154 held-out concept-target positions rises 2.8 times, from 0.10% under NTP post-training to 0.27% (perplexity, its inverse, from 1,015 to 369).

Conceptizer also gives the contextual synonyms $w_1, \ldots, w_m$ the most probability against the 3 baselines in every model (seed 0; Appendix F). Conceptizer's *judged-synonym probability*, the total probability of a position's contextual synonyms $w_1, \ldots, w_m$ without the observed word w_0, is in geometric mean 2.2 to 3.1 times NTP post-training's, 1.5 to 2.0 times Randomized synonyms' and 2.4 to 5.6 times the base model's (for Llama 1B, 0.464% against 0.216%, 0.313% and 0.192%). We also tried an alternative, *Weighted NTP*, which puts the concept loss's weight λ on the observed word w_0 alone: compared with Conceptizer, it raises the observed word w_0's probability further and reaches comparable downstream performance, but gives the contextual synonyms $w_1, \ldots, w_m$ less probability and has a higher global perplexity.

On content-word positions, Randomized synonyms has the lowest perplexity in 8 of 9 models, and Conceptizer in Falcon3 7B (Fig. 4). This lead comes from content words that split into several tokens. Randomized synonyms raises the probability of their first tokens that belong to many concepts (*un-*, *re-*) but gives the gain back on the rest of the word (seed 0; Appendix F). Conceptizer comes at a modest cost in global perplexity, smaller than Randomized synonyms' in every model (for Llama 1B, 19.31 against 17.85 and 19.88; Fig. 4).

6.2 Downstream Evaluations

To assess the impact of concept training on downstream performance, we use 3 task groups, each weighting its tasks equally (splits and metrics in Appendix D). **Knowledge** is 4 science-question tasks, ARC-Challenge, ARC-Easy (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018) and SciQ (Welbl et al., 2017), scored by zero-shot answer likelihood (Gao et al., 2024). **Sentiment** is 9 classification tasks of sentiment, emotion and toxicity from MTEB (Muennighoff et al., 2023; Enevoldsen et al., 2025): IMDb (Maas et al., 2011), ToxicConversations (cjadams et al., 2019; Borkan et al., 2019), TweetSentimentExtraction (Maggie et al., 2020), Emotion (Saravia et al., 2018), ToxicChat (Lin et al., 2023), PoemSentiment (Sheng and Uthus, 2020), FrenkEn (Ljubešić et al., 2019), YelpReviewFull (Zhang et al., 2015) and AmazonPolarity (McAuley and Leskovec, 2013; Zhang et al.,

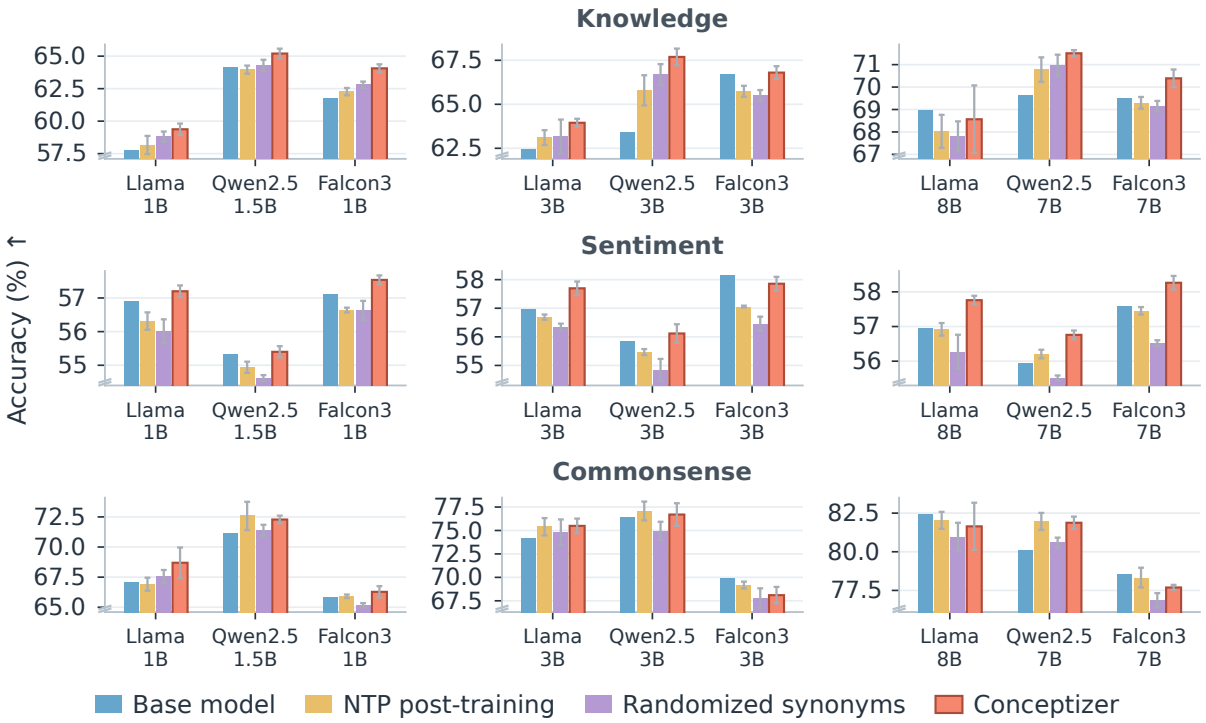


Figure 5: **Conceptizer leads on knowledge and sentiment tasks in 8 of 9 models each.** Commonsense is comparable. Means of seeds 0, 37 and 42; error bars: each method's 95% Student-t training-seed interval.

2015). A logistic-regression classifier trained on 8 to 128 labeled examples per label measures how well final-layer hidden states separate the labels. **Commonsense** is HellaSwag (Zellers et al., 2019), COPA (Roemmele et al., 2011) and WinoGrande (Sakaguchi et al., 2020), scored like knowledge.

Results. On knowledge tasks, Conceptizer has the best score against the 3 baselines in 8 of 9 models and outperforms them in 26 of 27 comparisons (9 models × 3 baselines; Fig. 5; Table 10). Its gains over NTP post-training, 0.54 to 1.90 points, have paired training-seed intervals (pointwise 95%, Student-t) above zero in all models but Llama 8B. Its gains over Randomized synonyms and the base model have intervals above zero in 7 models each. Per task, Conceptizer is best on OpenBookQA in all 9 models and on ARC-Challenge and ARC-Easy in 6 each (Table 13).

On sentiment tasks, Conceptizer has the best score against the 3 baselines in 8 of 9 models and outperforms them in 26 of 27 comparisons (Fig. 5; Table 11). Its gains over NTP post-training (0.46 to 1.01 points) and Randomized synonyms (0.78 to 1.74 points) have paired intervals above zero in all 9 models. Its gains over the base model have intervals above zero in 6 models. Per task, Conceptizer is best on TweetSentimentExtraction in all 9 models, on the 7 other tasks in 5 to 7 models each, and on YelpReviewFull in none (Table 15).

On commonsense tasks, the methods are comparable: Conceptizer, NTP post-training and the base model each have the best score in 3 models (Fig. 5; Table 12). Conceptizer is within 1.80 points of the best and above Randomized synonyms in every model. Against NTP post-training, its paired intervals include zero except in Llama 1B (+1.78 points, all from COPA's 100 items).

7 Conclusion

We introduced Conceptizer, a training framework that rewards meaning across the whole next-token distribution. Its fixed, model-agnostic concept vocabulary is built once from contextual synonyms and shared by every model through concept memberships, which standardizes the concept curation pipeline. Because earlier concept losses are special cases of its concept loss, Conceptizer also standardizes the concept training process. Against 3 baselines, across 3 model families and 3 scales, it gives the observed word and its contextual synonyms the most probability at held-out concept-target positions. It also leads on knowledge and sentiment tasks and remains comparable on commonsense, at a marginal cost in global perplexity.

Limitations

The candidate generators and the judge are open models, not frontier models, and are more prone to mistakes. Biases of these models and of WordNet can also pass into the synonym pools and the concepts built from them. First-token memberships can also link half-related words, such as *care* and *carelessness*. The judged-synonym probability is measured against synonyms kept by the same judge, so it is a reference point for how the training signal generalizes, not ground truth about meaning. By design, the concept target $\mathbf{q}_i$ can ask more of the closest concept $c^\star$ than the model concept distribution $\mathbf{p}_{c,i}$ can give, although the concept loss $\mathcal{L}_{\text{concept}}$ still asks to raise the logits of the tokens whose concepts are under-served on average (Appendix B). How much of each word's share ρ_j to send directly to the closest concept $c^\star$ (σ_j) deserves more careful experiments in future work. The hyperparameters follow a sweep (Appendix E) and the settings of previous work, and could be tuned more thoroughly. Concept-target positions are only 0.15 to 0.24% of training positions: about a third of positions are content words, only 3% of those are collected for labeling (Llama), and a target needs its observed word w_0 and at least one synonym to be single tokens, which most collected occurrences are not (Appendix C). Words that split into several tokens already have synonym pools and judge weights r_j, so expanding Conceptizer to multi-token words will unlock more coverage in future work without extra data curation. We study English text and models up to 8B parameters, which leaves larger models to future work.

Ethical Considerations

This work post-trains released LMs (Llama, Qwen2.5 and Falcon3) on public web text and builds its concept vocabulary with other released models (Llama-3.3-70B-Instruct and Qwen2.5-72B-Instruct, with gpt-oss-120B as the judge); all of these models may carry biases from their training data, which the post-trained models and the synonym pools can inherit. AI assistants supported research planning, implementation, code review, literature retrieval, figure preparation and manuscript editing; the authors made the scientific decisions, verified the sources and are responsible for the paper.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 Task 2: Semantic Textual Similarity, English, Spanish and Pilot on Interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 Task 10: Multilingual Semantic Textual Similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91, Dublin, Ireland. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic Textual Similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Yamen Ajjour, Henning Wachsmuth, Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. 2019. Data Acquisition for Argument Search: The args.me Corpus. In *KI 2019: Advances in Artificial Intelligence*, volume 11793 of *Lecture Notes in Computer Science*, pages 48–59. Springer.

Asaf Amrami and Yoav Goldberg. 2018. Word Sense Induction with Neural biLM and Symmetric Patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867, Brussels, Belgium. Association for Computational Linguistics.

Emanuele Bastianelli, Andrea Vanzo, Pawel Swietojanski, and Verena Rieser. 2020. SLURP: A Spoken Language Understanding Resource Package. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7252–7262, Online. Association for Computational Linguistics.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and

Siva Reddy. 2024. LLM2Vec: Large Language Models Are Secretly Powerful Text Encoders. In *First Conference on Language Modeling*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Sebastopol, CA.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2020. Overview of Touché 2020: Argument Retrieval. In *Working Notes of CLEF 2020 – Conference and Labs of the Evaluation Forum*, volume 2696 of *CEUR Workshop Proceedings*, Thessaloniki, Greece.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced Metrics for Measuring Unintended Bias with Real Data for Text Classification. In *Companion Proceedings of the 2019 World Wide Web Conference*, pages 491–500, San Francisco, CA, USA. ACM.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-Based *n*-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–480.

Natasha Butt, Ariel Kwiatkowski, Ismail Labiad, Julia Kempe, and Yann Ollivier. 2026. Soft Tokens, Hard Truths. In *International Conference on Learning Representations*, pages 114650–114675.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient Intent Detection with Dual Sentence Encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Xi Chen, Ali Zeynali, Chico Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1094–1106, Seattle, United States. Association for Computational Linguistics.

Jesús Cid-Sueiro. 2012. Proper losses for learning from partial labels. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

cjadams, Daniel Borkan, inversion, Jeffrey Sorensen, Lucas Dixon, Lucy Vasserman, and nithum. 2019. Jigsaw Unintended Bias in Toxicity Classification. Kaggle competition.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *Preprint*, arXiv:1803.05457.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Timothee Cour, Ben Sapp, and Ben Taskar. 2011. Learning from Partial Labels. *Journal of Machine Learning Research*, 12(42):1501–1536.

Lee R. Dice. 1945. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 26(3):297–302.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. CLIMATE-FEVER: A Dataset for Verification of Real-World Climate Claims. *Preprint*, arXiv:2012.00614.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2018. Token-level and sequence-level loss smoothing for RNN language models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2094–2103, Melbourne, Australia. Association for Computational Linguistics.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Gabriel Sequeira, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, and 67 others. 2025. MMTEB: Massive Multilingual Text Embedding Benchmark. In *International Conference on Learning Representations*, pages 101715–101771.

Matan Eyal, Shoval Sadde, Hillel Taub-Tabib, and Yoav Goldberg. 2022. Large Scale Substitution-based Word Sense Induction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4738–4752, Dublin, Ireland. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Falcon-LLM Team. 2024. The Falcon 3 Family of Open Models. Hugging Face blog, Technology Innovation Institute.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. 2023. MASSIVE: A 1M-Example Multilingual Natural Language Understanding Dataset with 51 Typologically-Diverse Languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4277–4302, Toronto, Canada. Association for Computational Linguistics.

Philip Gage. 1994. A New Algorithm for Data Compression. *The C Users Journal*, 12(2):23–38.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The Language Model Evaluation Harness. Zenodo.

Gregor Geigle, Nils Reimers, Andreas Rücklé, and Iryna Gurevych. 2021. TWEAC: Transformer with Extendable QA Agent Classifiers. *Preprint*, arXiv:2104.07081.

Joshua Goodman. 2001. Classes for Fast Maximum Entropy Training. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 561–564, Salt Lake City, UT, USA. IEEE.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. 2024. The Llama 3 Herd of Models. *Preprint*, arXiv:2407.21783.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2025. Training Large Language Models to Reason in a Continuous Latent Space. In *Second Conference on Language Modeling*.

John Hewitt, John Thickstun, Christopher Manning, and Percy Liang. 2023. Backpack Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9103–9125, Toronto, Canada. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *Preprint*, arXiv:1503.02531.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface Form Competition: Why the Highest Probability Answer Isn't Always Right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. Zenodo.

Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. 2015. CQADupStack: A Benchmark Data Set for Community Question-Answering Research. In *Proceedings of the 20th Australasian Document Computing Symposium*, pages 1–8, Parramatta, NSW, Australia. ACM.

Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Laya Iyer, Pranav Somani, Alice Guo, Dan Jurafsky, and Chen Shani. 2026. Beyond Tokens: Concept-Level Training Objectives for LLMs. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 457–474, Rabat, Morocco. Association for Computational Linguistics.

Paul Jaccard. 1912. The Distribution of the Flora in the Alpine Zone. *New Phytologist*, 11(2):37–50.

Yoon Kim and Alexander M. Rush. 2016. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. In *International Conference on Learning Representations*.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A Continuously Growing Dataset of Sentential Paraphrases. In *Proceedings of the 2017 Conference on*

Empirical Methods in Natural Language Processing, pages 1224–1234, Copenhagen, Denmark. Association for Computational Linguistics.

Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. In *Machine Learning Proceedings 1995*, pages 331–339, San Francisco, CA. Morgan Kaufmann.

Anne Lauscher, Ivan Vulić, Edoardo Maria Ponti, Anna Korhonen, and Goran Glavaš. 2020. Specializing Unsupervised Pretraining Models for Word-Level Semantic Similarity. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1371–1383, Barcelona, Spain (Online). International Committee on Computational Linguistics.

LCM team, Loïc Barrault, Paul-Ambroise Duquenne, Maha Elbayad, Artyom Kozhevnikov, Belen Alastruey, Pierre Andrews, Mariano Coria, Guillaume Couairon, Marta R. Costa-jussà, David Dale, Hady Elsahar, Kevin Heffernan, João Maria Janeiro, Tuan Tran, Christophe Ropers, Eduardo Sánchez, Robin San Roman, Alexandre Mourachko, and 2 others. 2024. Large Concept Models: Language Modeling in a Sentence Representation Space. *Preprint*, arXiv:2412.08821.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez. 2016. Semi-supervised Question Retrieval with Gated Convolutions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1279–1289, San Diego, California. Association for Computational Linguistics.

Yoav Levine, Barak Lenz, Or Dagan, Ori Ram, Dan Padnos, Or Sharir, Shai Shalev-Shwartz, Amnon Shashua, and Yoav Shoham. 2020. SenseBERT: Driving Some Sense into BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4656–4667, Online. Association for Computational Linguistics.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A Comprehensive Multilingual Task-Oriented Semantic Parsing Benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Zuchao Li, Rui Wang, Kehai Chen, Masao Utiyama, Eiichiro Sumita, Zhuosheng Zhang, and Hai Zhao. 2020. Data-dependent Gaussian Prior Objective for Language Generation. In *International Conference on Learning Representations*.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2024. Not All Tokens Are What You Need for Pretraining. In *Advances in Neural Information Processing Systems*, volume 37, pages 29029–29063. Curran Associates, Inc.

Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. 2023. ToxicChat: Unveiling Hidden Challenges of Toxicity Detection in Real-World User-AI Conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4694–4702, Singapore. Association for Computational Linguistics.

Yuliang Liu, Yunchong Song, Yixuan Wang, Kewen Ge, Alex Lamb, Qipeng Guo, Kai Chen, Bowen Zhou, and Zhouhan Lin. 2026. Next Concept Prediction in Discrete Latent Space Leads to Stronger Language Models. *Preprint*, arXiv:2602.08984.

Nikola Ljubešić, Darja Fišer, and Tomaž Erjavec. 2019. The FRENK Datasets of Socially Unacceptable Discourse in Slovene and English. In *Text, Speech, and Dialogue: 22nd International Conference, TSD 2019*, volume 11697 of *Lecture Notes in Computer Science*, pages 103–114, Cham. Springer.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Michal Lukasik, Himanshu Jain, Aditya Menon, Seungyeon Kim, Srinadh Bhojanapalli, Felix Yu, and Sanjiv Kumar. 2020. Semantic Label Smoothing for Sequence to Sequence Problems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4992–4998, Online. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Maggie, Phil Culliton, and Wei Chen. 2020. Tweet Sentiment Extraction. Kaggle competition.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. WWW'18 Open Challenge: Financial Opinion Mining and Question Answering. In *Companion Proceedings of the The Web Conference 2018*, pages 1941–1942, Lyon, France. International World Wide Web Conferences Steering Committee.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, Benjamin Bossan, and Marian Tietz. 2022. PEFT: State-of-the-art

Parameter-Efficient Fine-Tuning methods. GitHub repository.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 165–172, Hong Kong, China. ACM.

Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 Task 10: English Lexical Substitution Task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic. Association for Computational Linguistics.

Meta. 2024a. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. Meta AI blog.

Meta. 2024b. Llama 3.3 Model Card. GitHub, meta-llama/llama-models.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. MTEB: Massive Text Embedding Benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

James O'Neill, Polina Rozenshtein, Ryuichi Kiryo, Motoko Kubota, and Danushka Bollegala. 2021. I Wish I Would Have Loved This One, But I Didn't – A Multilingual Dataset for Counterfactual Detection in Product Reviews. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7092–7108, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

OpenAI. 2025. gpt-oss-120b & gpt-oss-20b Model Card. *Preprint*, arXiv:2508.10925.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 37, pages 30811–30849. Curran Associates, Inc.

Qwen Team. 2024. Qwen2.5 Technical Report. *Preprint*, arXiv:2412.15115.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. OpenAI technical report.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford, CA, USA.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740. AAAI Press.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized Affect Representations for Emotion Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean Voice Search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152, Kyoto, Japan. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Darsh Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. 2018. Adversarial Domain Adaptation for Duplicate Question Detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1056–1063, Brussels, Belgium. Association for Computational Linguistics.

Chen Shani, Jilles Vreeken, and Dafna Shahaf. 2023. Towards Concept-Aware Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13158–13170, Singapore. Association for Computational Linguistics.

Emily Sheng and David Uthus. 2020. Investigating Societal Biases in a Poetry Composition System. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 93–106, Barcelona, Spain (Online). Association for Computational Linguistics.

Gizem Soğancıoğlu, Hakime Öztürk, and Arzucan Özgür. 2017. BIOSSES: a semantic sentence similarity estimation system for the biomedical domain. *Bioinformatics*, 33(14):i49–i58.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Las Vegas, NV, USA. IEEE.

Jihoon Tack, Jack Lanchantin, Jane Dwivedi-Yu, Andrew Cohen, Ilia Kulikov, Janice Lan, Shibo Hao, Yuandong Tian, Jason Weston, and Xian Li. 2026. LLM Pretraining with Continuous Concepts. In *International Conference on Learning Representations*, pages 54214–54231.

Nandan Thakur, Luiz Bonifacio, Maik Fröbe, Alexander Bondarenko, Ehsan Kamalloo, Martin Potthast, Matthias Hagen, and Jimmy Lin. 2024. Systematic Evaluation of Neural Retrieval Models on the Touché 2020 Argument Retrieval Subset of BEIR. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1420–1430, Washington, DC, USA. ACM.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2020. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. *ACM SIGIR Forum*, 54(1):1–12.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the Best Counterargument without Prior Topic Knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251, Melbourne, Australia. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, and 9 others. 2020. CORD-19: The COVID-19 Open Research Dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Yida Wang, Yinhe Zheng, Yong Jiang, and Minlie Huang. 2021. Diversifying Dialog Generation via Adaptive Label Smoothing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3507–3520, Online. Association for Computational Linguistics.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing Multiple Choice Science Questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven

Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, and 12 others. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *Preprint*, arXiv:1609.08144.

Wei Xu, Chris Callison-Burch, and Bill Dolan. 2015. SemEval-2015 Task 1: Paraphrase and Semantic Similarity in Twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 1–11, Denver, Colorado. Association for Computational Linguistics.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. 2014. Extracting Lexically Divergent Paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2:435–448.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Christine Zhang, Dan Jurafsky, and Chen Shani. 2026. Learning Concepts, Not Tokens: Self-Supervised Semantic Alignment for Language Models. *Preprint*, arXiv:2603.29123.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. 2025. Soft Thinking: Unlocking the Reasoning Potential of LLMs in Continuous Concept Space. In *Advances in Neural Information Processing Systems*, volume 38, pages 168990–169012. Curran Associates, Inc.

A How the Concept Loss Acts on Each Word

Appendix A follows the concept loss $\mathcal{L}_{\text{concept}}$ at the running example, after "…practice the following sun", for Llama-3.2-1B-Instruct before the first optimizer step, when the model is still the base model. Table 1 gives the quantities of Section 4 for each kind of word. The pulls in Table 1 are $\sum_c \omega_{vc}\, q_i(c)/p_{\text{c},i}(c)$, the sum of the token v's uplift shares over all its concepts $\mathcal{H}(v)$, with the membership weights ω_{vc}, the concept target $\mathbf{q}_i$ and the model concept distribution $\mathbf{p}_{\text{c},i}$. The pushes are $p_{\text{w},i}(v)\,(\text{pull} - 1)$, where $\mathbf{p}_{\text{w},i}$ is the next-token distribution. A push is the change the concept loss asks for in the logit $z_i(v)$, the negative of the gradient in Equation 11. Table 1 gives it before the objective $\mathcal{L}$ weights each concept target $\mathbf{q}_i$ by $\lambda/|\mathcal{S}|$, the concept-loss weight over the number of concept-target positions (Equation 8). A concept's uplift is its target divided by its model probability, $q_i(c)/p_{\text{c},i}(c)$, and a token's logit $z_i(v)$ rises when its push is positive, that is, when the concept loss raises it, and falls when its push is negative.

What the cases show. The whole concept is pulled, whether or not a member word fits the sentence. The logits of *fortress* (4 concepts) and *bunker* (9), which do not fit the sentence, rise with those of *security* and the contextual synonyms $w_1, \ldots, w_m$.

The logits of tokens with fewer concepts than the uplift rise. For a token of concept #1450 with fewer than 13.1 concepts, #1450 alone adds more than 1 to its pull, so the logits of all 23 such tokens rise. The other 40 logits all fall except that of *protection*, a contextual synonym w_1 whose 21 other concepts also get part of the target $\mathbf{q}_i$ (at least 0.811% each, as #5336 in panel b) and lift its pull from 0.60 to 1.06.

A contextual synonym w_j is pulled through all its concepts $\mathcal{H}(w_j)$. The 6 concepts only *prevention* feeds each get 1.447% of the target $\mathbf{q}_i$ but only 0.157 to 0.217% of the model concept distribution $\mathbf{p}_{\text{c},i}$, so together they add 6.17 to its pull, almost 4 times what #1450 adds.

The logit of a synonym outside the closest concept $c^\star$ still rises. *Care* sends none of its 4.1% judge-weight share ρ_3 to #1450 and 0.091% to each of its 45 concepts (#3086, shared with *protection*, gets $0.811 + 0.091 = 0.902\%$). The base model's next-token distribution $\mathbf{p}_{\text{w},i}$ puts 0.97% on *care*, which gives each only about 0.022% of the model concept distribution $\mathbf{p}_{\text{c},i}$: on #2859 the target $\mathbf{q}_i$ asks 0.091% and gets 0.022%, an uplift of 4.2, and *care*'s pull is 3.39.

Concepts the sentence rules out get nothing. The observed word w_0 assigns all of its share ρ_0 to the closest concept $c^\star$ ($\sigma_0 = 1$). Of *safety*'s 8 other concepts (other than #1450), the 5 that con-

(a) Member words of the closest concept $c^\star$, #1450
$p_{\text{c},i}(c_{1450}) = 5.23\%$, $q_i(c_{1450}) = 68.7\%$: uplift 13.1

Word v	Concepts $\lvert\mathcal{H}(v)\rvert$	$p_{\text{w},i}$ (%)	Uplift share (×)	Pull (×)	Push (logit)
Fewer than 14 concepts					
safety	9	32.1	1.46	1.58	$+0.19$
security	9	0.07	1.46	1.57	$+0.0004$
fortress	4	<0.001	3.28	3.28	$+2\times10^{-7}$
bunker	9	<0.001	1.46	1.48	$+3\times10^{-8}$
14 or more concepts					
protection	22	32.1	0.60	1.06	$+0.02$
shield	53	0.014	0.25	0.30	-0.0001

(b) Other concepts in the target

Concept c	Word v ($\lvert\mathcal{H}(v)\rvert$)	$p_{\text{c},i}(c)$ (%)	$q_i(c)$ (%)	Uplift (×)	Uplift share (×)
Under-served					
#1379	prevention (8)	0.160	1.447	9.0	1.13
#2859	care (45)	0.022	0.091	4.2	0.09
Over-served					
#5336	protection (22)	5.042	0.811	0.16	0.007

(c) Words with no target

Word v	Concepts $\lvert\mathcal{H}(v)\rvert$	$p_{\text{w},i}$ (%)	Uplift share (×)	Pull (×)	Push (logit)
All concepts outside the target					
protective	7	1.8	none	0	-0.018
One over-served concept in the target					
habits	11	2.1	none	0.04	-0.020
In no concept					
-related	1 ($\perp$)	6.3	none	0	-0.063

Table 1: **What the concept loss $\mathcal{L}_{\text{concept}}$ asks of each word at the running example**, before the first optimizer step, where the model is the base model. $p_{\text{w},i}$ and $p_{\text{c},i}$: the next-token distribution and the model concept distribution (Equation 4), Fig. 2's p; q_i: the concept target (Equation 3), Fig. 2's q; Concepts: the word's number of concepts $\lvert\mathcal{H}(v)\rvert$ (in parentheses in panel b); Uplift share: the uplift of #1450 (panels a and c) or of the row's concept (panel b), divided by the word's number of concepts $\lvert\mathcal{H}(v)\rvert$. Words with no target: neither the observed word w_0 nor one of its contextual synonyms $w_1, \ldots, w_m$. Panel (a) splits at 14, the first whole number above the uplift 13.1. Pushes are on logits $z_i(v)$, not measured changes.

tain *protection* get part of the target $\mathbf{q}_i$, but the 3 that contain no contextual synonym w_j of this occurrence are outside it: #6298 {*deliverance*, *rescue*, *salvation*, ... }, #6387 {*lifelines*, *ropes*, *safetylines*, ... } and #6389 {*aid*, *safety-net*, *support*, ... }.

The logits of tokens in no concept fall. The pushes sum to zero (Appendix B), so every rise in a logit is paid for. Here the tokens in no concept, such as the word pieces *-related* and *-safe*, hold 20.9% and their logits all fall, because the concept target $\mathbf{q}_i$ gives the no-concept outcome $\perp$ nothing.

First-token memberships add concepts. *Care* joins 12 of its 45 concepts only through longer words that begin with its token, for example *careening*, *careworn*, *carefree* and *carelessness*.

The same example in the 3 families. Table 2 repeats the running example in each family's own training file. Its 4 words have the same numbers of concepts $|\mathcal{H}(v)|$ in all 3 tokenizers, except *care*, which is not in #1450. So the target $\mathbf{q}_i$ on #1450 is the same in all 3, 68.7%. Its *bound*, the most probability the model concept distribution $\mathbf{p}_{\mathrm{c},i}$ can put on #1450 (Appendix B), is also 50% in all 3. It is set by the tokens of #1450 in just 2 concepts: *safeg* in Llama and Qwen2.5 (Appendix C), *defences* and *fortifications* in Falcon3. What differs is how each base model spreads its probability, which changes every uplift and pull. In each family, the logit $z_i(v)$ of every token of #1450 with fewer concepts than #1450's uplift rises: 23 of 23 in Llama (uplift 13.1), 17 of 17 in Qwen2.5 (10.9) and 19 of 19 in Falcon3 (8.5).

Care has 45 concepts in Llama and Qwen2.5 but 44 in Falcon3, through first-token memberships. Falcon3 keeps *carefree* and *carelessness* as single tokens, so *care* loses the 3 concepts it reaches only through them (#839 {*carelessness*, *error*, *mistake*, ... }, #7049 {*carefree*, *casual*, *offhand*, ... } and #7095 {*calm*, *carefree*, *relaxed*, ... }). Falcon3 also splits *caretaking* into *care* and *taking*, not *caret* and *aking* as in Llama and Qwen2.5, which gives *care* 2 other concepts (#6682 {*caretaking*, *nursing*, *nurturing*, ... } and #7111 {*caretaking*, *maintenance*, *repair*, ... }).

	Llama	Qwen2.5	Falcon3
Tokenizer			
Concepts per member token	11.9	11.9	8.3
Tokens in #1450	63	63	67
Bound on #1450 (%)	50	50	50
Number of concepts $\lvert\mathcal{H}(v)\rvert$			
safety	9	9	9
protection	22	22	22
prevention	8	8	8
care	45	45	44
Base model: $p_{\mathrm{w},i}$ *(%)*			
safety	32.1	49.9	65.5
protection	32.1	16.2	12.9
prevention	1.24	0.04	1.54
care	0.97	0.34	0.29
$p_{\mathrm{c},i}(c_{1450})$ (%)	5.23	6.29	8.12
Target $q_i(c_{1450})$ (%)	68.7	68.7	68.7
Uplift on #1450 ($\times$)	13.1	10.9	8.5
Base model: pull ($\times$)			
safety	1.58	1.31	1.02
protection	1.06	1.33	1.42
prevention	7.87	166	5.24
care	3.39	10.4	11.3
After post-training: the 4 words' total $p_{\mathrm{w},i}$ *(%)*			
NTP post-training	75.8	67.4	76.6
Conceptizer	71.4	75.7	47.4

Table 2: **The running example keeps its target and its bound on #1450 in all 3 families, while each base model changes the uplifts and pulls.** The same training position with each family's different concept memberships $\mathcal{H}(v)$ (from its tokenizer) and different base model (Llama-3.2-1B-Instruct, Qwen2.5-1.5B-Instruct and Falcon3-1B-Instruct). Concepts per member token: the average number of concepts a token belongs to, $|\mathcal{H}(v)|$, over the tokens that belong to at least one concept. Bound on #1450: the most probability the model concept distribution $\mathbf{p}_{\mathrm{c},i}$ can put on #1450 (Equation 14). Symbols as in Table 1. The last 2 rows give the 4 words' combined probability under the seed-0 adapters at this training position, not held-out evidence.

B Derivations for the Concept Loss

Appendix B derives the gradient of the concept loss through the concept memberships, the target on the closest concept, the most probability the model concept distribution can put on a concept, and earlier losses as special cases. The implemented concept loss at position i is $\ell_i = -\sum_{c\in\mathcal{C}^+} q_i(c)\log\big(p_{\mathrm{c},i}(c)+\varepsilon\big)$ with $\varepsilon = 10^{-12}$, which keeps the logarithm finite when a concept gets no probability. With $\varepsilon = 0$, the mean of ℓ_i over the concept-target positions $\mathcal{S}$ is the concept loss $\mathcal{L}_{\text{concept}}$ (Equation 6). In an optimizer step with no concept-target positions, the concept loss is zero.

Gradient through concept memberships. For output tokens $u, v \in \mathcal{V}$, the softmax gives $\partial p_{\mathrm{w},i}(u)/\partial z_i(v) = p_{\mathrm{w},i}(u)\big(\mathbf{1}[u=v] - p_{\mathrm{w},i}(v)\big)$, where $\mathbf{1}[u=v]$ is 1 if $u=v$ and 0 otherwise, so by Equation 4

$$\frac{\partial p_{\mathrm{c},i}(c)}{\partial z_i(v)} = p_{\mathrm{w},i}(v)\big(\omega_{vc} - p_{\mathrm{c},i}(c)\big). \qquad (9)$$

Raising $z_i(v)$ thus raises $p_{\mathrm{c},i}(c)$ only for the concepts whose membership weight ω_{vc} exceeds their current probability $p_{\mathrm{c},i}(c)$. The exact derivative of the implemented loss ℓ_i is

$$\frac{\partial \ell_i}{\partial z_i(v)} = p_{\mathrm{w},i}(v) \sum_{c \in \mathcal{C}^+} q_i(c)\, \frac{p_{\mathrm{c},i}(c) - \omega_{vc}}{p_{\mathrm{c},i}(c) + \varepsilon}. \quad (10)$$

When $\varepsilon \to 0$ and $p_{\mathrm{c},i}(c) > 0$ wherever $q_i(c) > 0$, $\sum_c q_i(c) = 1$ turns Equation 10 into

$$p_{\mathrm{w},i}(v) \Big[1 - \sum_{c:\, q_i(c)>0} \frac{\omega_{vc}\, q_i(c)}{p_{\mathrm{c},i}(c)} \Big]. \quad (11)$$

The sum is the pull of v (Section 4), the average uplift of v's concepts $\mathcal{H}(v)$, since v's probability is split evenly over them, and the push $p_{\mathrm{w},i}(v)$ (pull $-$ 1) is minus this derivative. The loss ℓ_i therefore asks for v's logit $z_i(v)$ to rise exactly when its concepts are under-served on average. Written out, the push is v's *routed target*, the part of the target $\mathbf{q}_i$ routed to v, minus its probability $p_{\mathrm{w},i}(v)$,

$$p_{\mathrm{w},i}(v)\,(\text{pull} - 1) = \sum_{c:\, q_i(c)>0} q_i(c)\, \frac{\omega_{vc}\, p_{\mathrm{w},i}(v)}{p_{\mathrm{c},i}(c)} - p_{\mathrm{w},i}(v), \quad (12)$$

where each concept's target $q_i(c)$ is split among its tokens in proportion to their current probability within the concept, $\omega_{vc}\, p_{\mathrm{w},i}(v)/p_{\mathrm{c},i}(c)$. At the running example, *safety* supplies $32.1\%/9 = 3.57\%$ of $p_{\mathrm{c},i}(c_{1450}) = 5.23\%$, that is 68%, so it is routed 68% of the target $q_i(c_{1450}) = 68.7\%$, or 46.8%, and 50.7% in all once its 5 other concepts in the target are counted. Against its 32.1%, that is the push of $0.507 - 0.321 = +0.19$. At a given step the concept loss ℓ_i therefore asks the logits $\mathbf{z}_i$ for the same change as a token-level cross-entropy toward this routed target would, but the routed target changes with the model. Summing Equation 12 over $\mathcal{V}$, $\sum_v \omega_{vc}\, p_{\mathrm{w},i}(v) = p_{\mathrm{c},i}(c)$ leaves $\sum_c q_i(c) - 1 = 0$: the pushes sum to zero, so every rise in a logit is paid for by falls elsewhere. With $\varepsilon > 0$, the 1 in Equation 11 becomes $\sum_c q_i(c)\, p_{\mathrm{c},i}(c)/(p_{\mathrm{c},i}(c)+\varepsilon)$, slightly below 1, and each uplift becomes $q_i(c)/(p_{\mathrm{c},i}(c) + \varepsilon)$.

The target on the closest concept. By Equation 3, the target's mass on the closest concept $c^\star$ is

$$q_i(c^\star) = \sum_{j=0}^{m} \rho_j\, \kappa_j, \qquad \kappa_j = \sigma_j + (1 - \sigma_j)\, \omega_{w_j c^\star}, \quad (13)$$

where κ_j, the ratio of Fig. 2, is the fraction of word w_j's judge-weight share ρ_j that reaches $c^\star$, directly (σ_j) or through the concept memberships of w_j's token ($(1 - \sigma_j)\, \omega_{w_j c^\star}$): 1 for the observed word w_0 ($\sigma_0 = 1$), $\frac{1}{2} + \frac{1}{2}/|\mathcal{H}(w_j)|$ for a synonym whose token belongs to $c^\star$ ($\sigma_j = \frac{1}{2}$; $|\mathcal{H}(w_j)|$ is the number of concepts of w_j's token), and 0 for any other synonym ($\sigma_j = 0$ and $\omega_{w_j c^\star} = 0$).

The most probability the model concept distribution can put on a concept. By Equation 4, $\mathbf{p}_{\mathrm{c},i} = \sum_v p_{\mathrm{w},i}(v)\, \boldsymbol{\omega}_v$ is a probability-weighted average of the tokens' membership weights $\boldsymbol{\omega}_v = (\omega_{vc})_{c \in \mathcal{C}^+}$, so $\mathbf{p}_{\mathrm{c},i}$ never gives a concept more than its largest membership weight, $p_{\mathrm{c},i}(c) \le \max_v \omega_{vc}$, with equality only when all probability is on such tokens. For the closest concept $c^\star$,

$$p_{\mathrm{c},i}(c^\star) \le \frac{1}{\min_{v:\, c^\star \in \mathcal{H}(v)} |\mathcal{H}(v)|}, \quad (14)$$

the inverse of the fewest concepts that any token of $c^\star$ belongs to, below 1 whenever every token of $c^\star$ also belongs to another concept. At the running example, every token of #1450 belongs to at least 2 concepts, so the model concept distribution $\mathbf{p}_{\mathrm{c},i}$ never gives #1450 more than 50%, against the 68.7% the target $\mathbf{q}_i$ asks (Table 2).[2] With $\varepsilon = 0$, ℓ_i is the target's entropy plus $D_{\mathrm{KL}}(\mathbf{q}_i \| \mathbf{p}_{\mathrm{c},i})$, so the loss reaches its minimum, the target's entropy, only at $\mathbf{p}_{\mathrm{c},i} = \mathbf{q}_i$. A target that asks its closest concept $c^\star$ for more than the bound therefore keeps the divergence above zero for every model, and training can move toward the target but never reach it. The bound concerns one coordinate, so meeting it does not make a target attainable. Many training targets ask their closest concept $c^\star$ for more than this bound: 57.9% of Llama's, 58.2% of Qwen2.5's and 36.4% of Falcon3's.

Earlier losses as special cases. With $\varepsilon = 0$, $\ell_i = -\sum_c q_i(c) \log p_{\mathrm{c},i}(c)$ rests on 3 choices: the concept vocabulary $\mathcal{C}$, each token's concepts $\mathcal{H}(v)$, which fix the membership weights ω_{vc}, and the target $\mathbf{q}_i$. Let $T_i \subseteq \mathcal{V}$ be the tokens an earlier method rewards at position i. Three settings of these choices recover known losses. **Next-token prediction.** Give every token a concept of its

[2] The bound is on the concept, not on a token: *safety*'s routed 50.7% above is a target for its token, which gives #1450 only 1/9 of its probability, so even all of the probability on *safety* would give #1450 just 11.1%. Only the tokens of #1450 in just 2 concepts, such as *safeg* for Llama, can bring it to 50% (Appendix A).

own, $\mathcal{H}(v) = \{v\}$, so $\omega_{vc} = \mathbf{1}[c = v]$ and $p_{c,i}(v) = p_{w,i}(v)$, and put the whole target on the observed token g_i, $\mathbf{q}_i = \mathbf{e}_{g_i}$:

$$\ell_i = -\log p_{w,i}(g_i). \tag{15}$$

Next-concept prediction (Iyer et al., 2026). Keep these one-token concepts and spread the target evenly over T_i, $\mathbf{q}_i = |T_i|^{-1} \sum_{t \in T_i} \mathbf{e}_t$:

$$\ell_i = -\frac{1}{|T_i|} \sum_{t \in T_i} \log p_{w,i}(t), \tag{16}$$

the negative of their next-concept-prediction objective (their Section 2.2.2), which rewards each token of T_i on its own. The match is exact when every completion is a single token: their loss also scores multi-token completions over their whole token span (their Appendix D), which a loss at one position cannot express. Their data augmentation, one NTP example per token of T_i, averages to the same loss. **Pooled synonyms** (Zhang et al., 2026). Let T_i hold the observed word w_0 and its contextual synonyms $w_1, \ldots, w_m$, give them one concept c_T of their own, $\mathcal{H}(t) = \{c_T\}$ for every $t \in T_i$, so $\omega_{tc_T} = 1$ and $p_{c,i}(c_T) = \sum_{t \in T_i} p_{w,i}(t)$, and put the whole target on it, $\mathbf{q}_i = \mathbf{e}_{c_T}$:

$$\ell_i = -\log \sum_{t \in T_i} p_{w,i}(t), \tag{17}$$

their concept loss (their Section 3.3), which rewards the pooled probability. The identities concern the concept loss alone: Zhang et al. interpolate it with NTP at content words and switch it off once T_i holds 60% of the probability, while Conceptizer adds it to NTP and retention, $\mathcal{L}_{\text{NTP}}$ and $\mathcal{L}_{\text{retention}}$ (Equation 8).

C Data Preparation Details

Appendix C gives the exact rules of Section 3 in the order the data pass through them. Occurrences of content words are collected, and their synonym candidates are proposed and judged. A sample of the judged occurrences builds the concept vocabulary, which concept memberships link to each tokenizer. Each occurrence that passes 6 rules then gets a concept target inside the training and held-out files.

Collection. Occurrences come from English-source FineWeb-Edu documents (release v1.4.0, its sample-100BT subset) with a language score of at least 0.90 (FineWeb-Edu's fastText confidence that a document is English) and at most 1,500,000 characters. Heuristic math and code filters keep the judged sentences to ordinary prose. They apply to each document's first 2,000 characters and to each sentence, and exclude text shorter than 20 characters, with more than 25% non-alphabetic, non-space characters or more than 15% digits, or matching code or LaTeX patterns. The filters act on collection only, so training windows, taken from the full documents, can still contain math or code. spaCy 3.8.14 with `en_core_web_sm` 3.8.0 tags the content words: alphabetic non-stopword nouns, verbs and adjectives, which leaves out proper nouns. A word is its lowercased spaCy lemma, so *Edible* and *edible* are one word.

Candidates. The 3 sources of Section 3 are set to cast a wide net, since the judge removes what does not fit. Llama-3.3-70B-Instruct reads the sentence with the first occurrence of the word's spelling marked. Qwen2.5-72B-Instruct sees no sentence, so it responds once per word, and the response is reused for all of the word's occurrences (prompts in Appendix H). Both draw 5 samples at temperature 1.0 and seed 0, with at most 128 and 192 tokens per response, respectively. Additionally, WordNet (Miller, 1995), read through NLTK (Bird et al., 2009), adds the lemma names (synonyms) of all the word's senses under its POS, without multiword phrases. It has such synonyms for 20.7% of the collected words, which hold 31.4% of the occurrences. The lowercased union of the 3 sources removes duplicates and records each candidate's sources. Falcon3's family proposed no candidates, yet in all 3 Falcon3 models Conceptizer has the lowest concept-target perplexity and the best knowledge score against the 3 baselines, and the best sentiment score in 2 (Fig. 4; Tables 10 and 11).

The judge. The LLM judge, gpt-oss-120B, narrows the candidates to the contextual synonyms. It reads the same marked sentence with the word, its POS and the candidates in alphabetical order (prompt in Appendix H), at temperature 0.7 with high reasoning effort, 3 times per occurrence. An occurrence is removed when at least 2 of its 3 answers mark the word as a name or code. Table 3 lists the judge's answers for the 25 occurrences of *safety*, with the judge weights of Equation 1.

Concept-construction sample. The concept vocabulary is built from a fixed sample of judged occurrences, the *concept-construction sample*, cho-

	Sentence (excerpt)	Candidates	*protection*	*prevention*	*care*	*security*
1	. . . child *safety* seats and bicycle helmets	39	3/3	not proposed	0/3	3/3
2	. . . follow certain *safety* precautions	44	3/3	0/3	0/3	3/3
3	. . . highway *safety* was a top concern	39	3/3	not proposed	0/3	3/3
4	. . . concerns about pilot and bird *safety*	39	3/3	not proposed	0/3	3/3
5	For the *safety* of the birds, we cannot. . .	41	3/3	not proposed	1/3	3/3
6	. . . related to *safety* and herbal medicines	41	3/3	not proposed	0/3	3/3
7	. . . *safety* assessment of herbal medicines	43	2/3	not proposed	0/3	2/3
8	. . . for *safety* assessment apply to herbal. . .	41	2/3	not proposed	0/3	2/3
9	. . . *safety* assessment exist in their country	41	3/3	not proposed	0/3	3/3
10	Regulatory requirements for *safety*. . .	41	1/3	not proposed	0/3	1/3
11	. . . requirements for *safety* assessment of. . .	41	1/3	not proposed	0/3	1/3
12	. . . requirements for *safety* assessment.	41	1/3	not proposed	0/3	0/3
13	. . . for *safety* assessment detailed above	41	2/3	not proposed	0/3	2/3
14	. . . mechanism for *safety* requirements	40	3/3	not proposed	0/3	3/3
15	. . . follow good summer *safety* habits. . .	43	3/3	2/3	0/3	3/3
16	. . . but *safety* comes first when enjoying. . .	41	3/3	not proposed	0/3	3/3
17	Keep the following *safety* precautions. . .	47	3/3	0/3	0/3	3/3
18	**. . . practice the following sun *safety* tips. . .**	**43**	**3/3**	**2/3**	**1/3**	**0/3**
19	. . . Practice good summer *safety*	43	3/3	1/3	0/3	2/3
20	Talk to your kids about school *safety*	39	3/3	not proposed	0/3	3/3
21	Contemporary *safety* principles of. . .	40	3/3	1/3 (added)	0/3	3/3
22	. . . multiple layers of *safety* systems. . .	40	3/3	not proposed	0/3	3/3
23	. . . capabilities and *safety* profiles than. . .	39	2/3	not proposed	0/3	0/3
24	. . . to meltdown and operating *safety*	39	3/3	not proposed	0/3	2/3
25	PIUS (process inherent ultimate *safety*)	39	3/3	not proposed	0/3	3/3
	Kept in / proposed in, k_j/n_j		25/25	4/6	2/25	22/25
	Judge weight $r_j = (k_j + 1)/(n_j + 2)$		$26/27 = 0.963$	$5/8 = 0.625$	$3/27 = 0.111$	$23/27 = 0.852$

Table 3: **The judge's answers behind the running example.** The 25 collected occurrences of *safety* and the number of candidates the 3 sources proposed. Each cell counts how many of the judge's 3 answers kept the word. Not proposed marks a word that was not a candidate, and (added) the one occurrence where the judge added *prevention* itself. Bold: the running example of Figs. 2 and 3. Rows 11 and 12, and 21 and 22, are 2 occurrences of *safety* in one sentence each, judged separately.

sen to hold words with several senses. It takes words with at least 15 occurrences in which the judge kept a candidate, and whose synonym pools form at least 2 groups when pools with a Jaccard similarity of at least 0.25 are linked. Words with the most such occurrences enter first, with ties broken by file order, until the sample holds 150,000 occurrences; all 6,000 that enter have the maximum of 25. Their sentences are tokenized and truncated at 256 tokens; the 81 occurrences in which no token of the sentence covers the word are dropped, and the other 149,919, from 17,326 documents, form the concept-construction sample. Two alignment checks then leave out 2,262 occurrences whose tokens do not spell the word and 2,423 that open their sentence. Finally, occurrences in 60% of the 17,326 documents build the concepts, and the other 40% are held out from construction. A word with fewer than 5 remaining occurrences in the 60% builds no concept. This leaves the 88,367 occurrences of 5,939 words of Section 3.

Senses and concepts. Section 3 builds the concepts in 2 steps. First, within a word, average linkage groups synonym pools into senses by their Dice similarity, comparing kept words after folding inflections (WordNet `morphy` under the word's POS, then the other POS, then suffix rules for `-ies`, `-ing`, `-ed` and plurals), so *rising* and *grown* compare as *rise* and *grow*. A sense's final pool keeps the surface forms: its occurrences' kept words and the word as written in each occurrence.

Second, across words, 2 senses form a *candidate pair* when their final pools share at least 2 surface forms and reach the Jaccard threshold. Pairs are taken in descending similarity, and complete linkage merges 2 groups only when every pair between them qualifies, which keeps chains of partial overlaps from joining unrelated words. Senses of the same word stay in different concepts, since they are different meanings of that word. Only words in at most 200 final pools are used to look up candidate pairs, which keeps the search for pairs tractable,

though every word counts in the similarity. Word lookup applies Unicode normalization, folds dash variants and removes case duplicates within a concept.

Concepts are numbered #0 to #9060 by first appearance in the concept-construction sample, and they have 2 to 169 words (median 20). Concept #1450 merges a sense of *bulwark* (11 occurrences, 44 words) with one of *defence* (14 occurrences, 53 words), whose final pools share 25 of their 72 words, among them *protection* and *shield*. It also contains *safety*, kept for *defence* in 3 occurrences, and loose member words such as *moat*, kept in 1. Appendix E varies the Dice and Jaccard thresholds and the linkage.

Concept memberships. Each tokenizer reaches the concepts through concept memberships (Section 3). A word's token is the first token of the word written with a leading space, as it appears after another word (for Llama, *safety* is one token, `␣safety`, but 2, *s* and *afety*, without the space), and the single-token checks use the same form. The 49,387 distinct member words in the concept vocabulary give 17,938 *member tokens* for Llama, 17,870 for Qwen2.5 and 27,430 for Falcon3, because many words share a first token: *care*'s token begins 8 member words (*care*, *careen*, *careened*, *careening*, *careens*, *carefree*, *carelessness* and *careworn*). All 9,061 concepts have a member token in each family, and a single-token member word in 8,926 concepts for Llama, 8,926 for Qwen2.5 and 9,029 for Falcon3.

Closest concept $c^\star$. Each concept target $\mathbf{q}_i$ sends the observed word w_0's whole share ρ_0 to its closest concept $c^\star$ (Section 3). Among the concepts that list the observed word w_0, $c^\star$ is the one containing the most contextual synonyms $w_1, \ldots, w_m$. If several concepts tie for the most, the lowest-numbered one becomes $c^\star$. Ties occur in 33% of Llama's and Qwen2.5's training targets and 27% of Falcon3's: at the running example, #1450 and #2628 each contain 2 synonyms, and the lower-numbered #1450 becomes $c^\star$. The same occurrence gets the same closest concept $c^\star$ from Llama's and Qwen2.5's tokenizers in 99.98% of the training targets they share and from all 3 tokenizers in 90.06%. Falcon3 keeps more synonyms as single tokens, which can move $c^\star$: *citizens* takes #1571 for Llama and Qwen2.5 but #2203 for Falcon3, which also keeps *citizenry* as one token. The next paragraph covers 2 special cases in which counting contextual synonyms cannot choose $c^\star$.

Token owners. In 2 special cases, counting contextual synonyms $w_1, \ldots, w_m$ cannot choose the closest concept $c^\star$. In the *first special case*, w_0 is listed in several concepts, but none of them contains a contextual synonym (7% of Llama's and Qwen2.5's training targets, 6% of Falcon3's). In the *second special case*, w_0 is not listed in any concept, and its token belongs to concepts only as the first token of longer member words (0.3%, 0.3% and 0.2%). For example, no concept lists *uninstall*, but #6852 lists *uninstalling*, which begins with the token of *uninstall* in all 3 tokenizers.

For the 2 special cases, each member token v, the first token of at least one member word, has an *owner*: one of its concept memberships $\mathcal{H}(v)$, designated in advance. In the first special case, the owner of w_0's token becomes the closest concept $c^\star$ if that owner lists w_0. Otherwise, the lowest-numbered concept that lists w_0 becomes $c^\star$. In the second special case, the owner of w_0's token always becomes $c^\star$, and rule 6 of "Which occurrences get a target" decides whether the target is kept. For example, the token of *uninstall* belongs only to #6852, through *uninstalling*, so #6852 owns it and becomes $c^\star$ for every occurrence of *uninstall*.

Owners are set by 2 rules, applied in order. The first rule gives each token to the lowest-numbered of its concept memberships $\mathcal{H}(v)$. However, alone, it would leave a concept owning no token whenever all of its member tokens also belong to lower-numbered concepts, and the second special case could never choose that concept. For example, the first rule gives the tokens of *ages* and *years* to #265 {*aeon*, *age*, *ages*, *centuries*, …, *years*}, leaving #267 {*ages*, *years*} owning none.

Therefore, the second rule lets each concept left owning no token take one of its member tokens from that token's owner under the first rule, as long as that owner keeps at least one other token. In the example, #267 takes the token of *years* from #265. In the end, #267 owns 1 token, and #265 owns 10 for Llama. The first rule gave 21 tokens to #265, and the second rule then moved 11 of them to 11 different concepts such as #267, one each.

The second rule changes the owner of 25.9% of Llama's member tokens, 26.1% of Qwen2.5's and 12.7% of Falcon3's. After both rules, every concept is the owner of at least one token except 126 of the 9,061 (1.4%) for Llama, 127 (1.4%) for Qwen2.5 and 38 (0.4%) for Falcon3, because

each of their member tokens is its owner's only token. #3248 {*gullet*, *larynx*, *pharynx*, *throat*} is one: each of its 4 member tokens is the only token of a different owner, such as #433 {*airway*, *throat*, *trachea*, *windpipe*, ... }, which owns only the token of *throat*. Owners matter only in the 2 special cases: every concept, owned or not, gets probability through all of its member tokens, and a concept that owns no token can still become $c^\star$ by counting contextual synonyms.

Which occurrences get a target. The concept loss reads one next-token distribution, so a concept target needs an observed word and synonyms that are single tokens, and a closest concept that contains the token that occurs. A collected occurrence therefore gets a concept target only if it passes 6 rules. The first 3 check the occurrence, and the last 3 build and check the target.

(1) Single tokens, not a name or code. The observed word w_0 is a single whole-word token, at most 1 of the judge's 3 answers marks it as a name or code, and at least one kept synonym is a single token.

(2) Concept memberships. The observed token w_0 belongs to a concept with fewer than 200 member words, and a single-token synonym belongs to a concept.

(3) Whole words, no negating prefix. The observed word w_0 starts and ends at word boundaries in the document, so its token is the whole word as written, and does not follow a negating prefix joined by a hyphen or dash, which would invert its meaning (*non-judgmental* never gives *judgmental* a target).

(4) At most 10 synonyms. Multi-token synonyms are dropped, and the rest are ranked by how many answers kept them and limited to 10 before judge weights r_j are attached.

(5) The observed token in $c^\star$. The exact observed token w_0 at that position belongs to the closest concept $c^\star$, so the target's largest share lands on a concept that the observed token gives probability to.

(6) Lexical support. In the second special case of the token owners, w_0 is listed in no concept, and the owner of its token becomes $c^\star$ only because that token is the first token of a member word. If no synonym is listed in $c^\star$ either, the target needs *lexical support*: a synonym with the same WordNet lemma as a member word of $c^\star$, neither word having any other lemma, and no lemma in common with w_0. For example, the token of baking *flour* is the first token of *flourishes*, so in Llama and Qwen2.5 it is owned by an outbreak concept (#6109 {*bloom*, *burst*, *flourishes*, ... }). None of *flour*'s synonyms (*cover*, *dust*, *powder*, *coat*, *sprinkle*) has the lemma of a member of #6109, so the target is removed. This rule removes about 9 in 10 such occurrences (Table 4), but it does not certify the sense of those it keeps.

A training target ends with 1 to 10 contextual synonyms $w_1, \ldots, w_m$, a median of 5 for Llama and Qwen2.5 and 6 for Falcon3: 1 synonym in 10.4% of Llama's targets, 10.5% of Qwen2.5's and 7.3% of Falcon3's, and the cap of 10 in 15.0% of Llama's, 15.0% of Qwen2.5's and 23.6% of Falcon3's. A synonym whose token belongs to no concept stays in the target $\mathbf{q}_i$, and its whole share ρ_j goes to the no-concept outcome $\perp$. Table 4 counts each rule's removals.

Documents and windows. Training reads running text: *training windows* of at most 256 tokens from the documents that hold collected occurrences. A document is tokenized whole and divided into consecutive windows. A document's final piece shorter than 32 tokens is dropped rather than trained as a nearly empty window, and so is a window that holds no collected occurrence. Windows carry no beginning-of-sequence token, so a window's first token is never predicted. The training files hold 814,799 windows for Llama, 820,136 for Qwen2.5 and 819,697 for Falcon3, and the 50,000 optimizer steps read the first 800,000 once, in one fixed random order of documents (each document's windows consecutive) shared by every method and seed.

Hashing each document's identifier holds out a random but fixed 10% of the documents for the held-out evaluation of Section 6, half for development and half for final evaluation. The 17,326 documents of the concept-construction sample are never held out, so no held-out text helped build a concept. A held-out occurrence gets a target as often as a training occurrence in documents outside the concept-construction sample (Llama 6.3% against 6.4%, Falcon3 12.2% against 12.4%). Held-out files keep the content-word positions that get no target, for the content-word perplexity.

What the files hold. For Llama, the training windows hold 196.8 million predicted positions, and 32% of them are content words (counted in every 400th window). Of these content words, 2,088,585

	Llama	Qwen2.5	Falcon3
Judged occurrences, all documents	2,327,384	2,327,384	2,327,384
outside the training documents' windows	236,040	236,001	236,106
Located in training documents' windows	2,091,344	2,091,383	2,091,278
in discarded short document ends	2,759	2,735	2,802
In kept training windows	2,088,585	2,088,648	2,088,476
Removed, in the order the checks apply			
Observed word w_0 splits into several tokens	1,598,264	1,599,645	1,270,830
Judge marks a name or code	37,243	36,899	56,601
No candidates	2,692	2,687	5,290
No kept synonym	28,739	28,495	54,519
No single-token synonym	36,598	36,478	51,664
Observed token w_0 in no concept	46,632	46,254	119,816
Negating prefix	6,294	6,275	7,735
Synonyms in no concept	3,014	3,005	6,192
Observed token w_0 not in the closest concept $c^\star$	2,602	2,586	4,007
Lexical-support check	6,606	6,591	7,738
Not a whole word in the document	21,495	21,506	28,526
Concept-target positions	**298,406**	**298,227**	**475,558**
fraction of occurrences in kept windows (%)	14.3	14.3	22.8

Table 4: **From judged occurrences to concept targets.** Each check applies to the occurrences that passed the earlier ones. The rows follow the order of the checks, not the numbering of the 6 rules in "Which occurrences get a target" (Appendix C). Two further checks, for a concept of 200 or more words and for a closest concept $c^\star$ with no member token, removed none. Occurrences outside the training documents' windows include those in the development and final held-out documents (164,408 / 164,411 / 164,458 for Llama / Qwen2.5 / Falcon3).

are collected occurrences (3.3%), and 76.5% of the collected occurrences have an observed word w_0 that splits into several tokens. The 298,406 that get concept targets are 0.15% of all positions. Training targets have 7,875 distinct closest concepts $c^\star$ for Llama, 7,875 for Qwen2.5 and 8,464 for Falcon3, and held-out targets 2,523 for Llama, 2,523 for Qwen2.5 and 3,648 for Falcon3.

Known errors. The lexical-support check (rule 6 of "Which occurrences get a target") removes some targets whose closest concept $c^\star$ is wrong. A correct removal: in Falcon3, the token of the musical *viol* is the first token of *violator* and belongs only to 2 offender concepts (#2641 and #5887), so its target, which went to #2641, is removed. However, the lexical-support check can also remove useful targets and keep wrong senses: *matt*, a dull finish, reaches a carpets concept (#9016 {*carpets*, *mats*, *rugs*, ...}) through the first token of *mattings*, and keeps its target in all 3 families because its synonym *mat* has the same lemma as the member *mats*. Wrong senses also pass where the check does not apply, because the observed word is itself a member word: in a *light water* reactor, where *light* means ordinary rather than heavy water, Falcon3's target for *light* still goes to an illumination concept (#4930 {*illuminate*, *light*, *lit*, ...}). A misspelling can also become a member word: the judge kept *safeguat*, whose first token makes *safeg* the only token of #1450 in just 2 concepts for Llama and Qwen2.5, and so sets #1450's 50% bound (Appendix B). These are examples, not an accuracy estimate.

D Training and Evaluation Settings

Appendix D lists the post-training and evaluation settings in Table 5. In every post-trained method the retention loss $\mathcal{L}_{\text{retention}}$ acts only at the other positions $\mathcal{U}$ (Equation 7), so NTP post-training trains the concept-target positions $\mathcal{S}$ with next-token prediction alone. On the same GPU in the same job, Conceptizer trains 5 to 8% longer than NTP post-training for the 1B and 1.5B models and 1 to 4% longer for the 3B to 8B models.

COPA. COPA is scored on its 100 SuperGLUE validation questions (Wang et al., 2019), which weigh as much as the larger tasks.

The sentiment classifier. MTEB's logistic-regression classifier (scikit-learn 1.8.0 (Pedregosa et al., 2011) defaults, at most 100 iterations; MTEB 2.18.6 `abstasks/classification.py`) is trained on the embeddings of 8 to 128 training examples per label, MTEB's setting for each task (IMDb 8, ToxicConversations 16, TweetSentimentExtraction

Setting	Value
Post-training	
Adapter	LoRA, rank 16, alpha 32 (updates scaled by alpha/rank $= 2$), dropout 0.05, no bias terms; the only trained parameters
Adapted layers	every linear attention and feed-forward projection (PEFT all-linear; Mangrulkar et al., 2022); output head excluded
Precision	bfloat16 model weights
Hardware	1 NVIDIA H100 per run; a 50,000-step run took about 7 to 9 hours of job time for the smallest models and about 20 hours for Llama 8B
Software	PyTorch 2.6.0 (Paszke et al., 2019), Transformers 5.0.0 (Wolf et al., 2020) and PEFT 0.19.1 for post-training and evaluation; NLTK 3.9.2 for the concept vocabulary
Optimizer	AdamW (Loshchilov and Hutter, 2019), betas (0.9, 0.999), epsilon 10^{-8}, weight decay 0.01; gradient norm clipped at 1.0
Learning rate	7×10^{-5}, as in Zhang et al. (2026); linear warmup over 100 steps, then cosine decay (Loshchilov and Hutter, 2017) to zero at step 50,000
Steps and batch	50,000 optimizer steps of 16 windows of at most 256 tokens, no gradient accumulation; the final checkpoint is evaluated
Loss weights	$\lambda = 0.6$ (concept), $\beta = 0.5$ (retention); $\varepsilon = 10^{-12}$ inside the concept logarithm
Training seeds	0, 37 and 42 (Weighted NTP: seed 0, the smallest model of each family); a seed sets the LoRA initialization and dropout, and every seed reads the same windows in the same order
Knowledge and Commonsense	
Harness	lm-evaluation-harness 0.4.10 (Gao et al., 2024); each task's native prompt, zero-shot, no chat template; the answer with the highest likelihood is chosen, nothing is generated
Knowledge	test splits of ARC-Challenge (1,172 questions), ARC-Easy (2,376), OpenBookQA (500) and SciQ (1,000); acc_norm; SciQ's prompt includes its supporting passage
Commonsense	validation splits of HellaSwag (10,042 questions; acc_norm), COPA (100; acc) and WinoGrande (1,267; acc)
Sentiment and MTEB	
Embeddings	MTEB 2.18.6; final-layer hidden states averaged over non-padding tokens in float32 (mean pooling, compared for decoder models by BehnamGhader et al., 2024); inputs truncated at 256 tokens; batch 32; no instruction prompt; no L2 normalization
Sentiment	9 classification tasks (Section 6), the 6 outside the 40-task suite in MTEB's .v2 versions; mean test-split accuracy of MTEB's logistic-regression classifier over 10 draws of training examples (below)
MTEB	the 40 tasks of Appendix G; each task's recorded score
Group scores	equal weight for every task of a group

Table 5: **Post-training and evaluation settings.** One recipe for all 9 models and every post-trained method, with nothing tuned per model or size.

32, Emotion 16, ToxicChat 8, PoemSentiment 8, FrenkEn 8, YelpReviewFull 128, AmazonPolarity 8).

The word-level comparison of Fig. 3. The figure reads the seed-0 Conceptizer and NTP post-training adapters of Llama 1B at the running example's training position with its full 256-token window: the probabilities of the single tokens of *safety* (ID 7296), *protection* (9313), *prevention* (27344) and *care* (2512), and of *habits* (26870), *related* (5552) and *screen* (4264), the 3 largest absolute probability decreases among 14,555 lowercase single-token member words of the concept vocabulary $\mathcal{C}$. Logits $\mathbf{z}_i$ are bfloat16 on CPU and the softmax is float64 over all 128,256 output tokens, so tokens whose logits round to the same bfloat16 value get the same probability.

Licenses and intended use. FineWeb-Edu is released under ODC-By 1.0 and WordNet 3.0 under Princeton's WordNet license. The post-trained models, the candidate generators and the judge are used under their own licenses: the Llama 3.1, 3.2 and 3.3 Community Licenses, Qwen's licenses for Qwen2.5, TII's Falcon license for Falcon3 and Apache 2.0 for gpt-oss-120B. The benchmarks are used for evaluation only, under their own terms. Our release, code under MIT and data derived from FineWeb-Edu, these models' outputs and WordNet, keeps the terms of its sources and is intended for research on language-model training.

E Sensitivity to Construction and Training Settings

Appendix E shows how the concept vocabulary and the held-out perplexity change under other construction and training settings.

Construction settings. The construction sweeps rebuild the concept vocabulary from the concept-construction sample with one setting changed at a time (Table 6, top; Fig. 6). A higher Dice threshold splits words into more senses and so gives more concepts. The lowest tested value that gives the verb *light* 2 senses is 0.15, although some of its pools still mix the ignite and illuminate meanings. A higher Jaccard threshold merges fewer senses and so gives more and smaller concepts. At Dice 0.05, the lowest tested value that leaves no concept of 200 or more words is 1/3. At 0.25, one concept has 223 words. Single linkage chains senses

Construction	Concepts $\lvert\mathcal{C}\rvert$	Largest	$\geq$200	Senses of *light*
Dice threshold for senses				
0.02	7,128	169	0	1
0.05	7,611	169	0	1
0.10	8,284	169	0	1
0.15 (adopted)	9,061	169	0	2
0.20	9,878	192	0	2
Jaccard threshold for concepts				
0.25	6,679	223	1	1
1/3 (adopted)	7,611	169	0	1
0.50	8,422	134	0	1
Linkage for concepts				
Single	7,174	411	17	1
Complete (adopted)	7,611	169	0	1

	Held-out perplexity ↓		
Training	Concept-target	Content-word	All tokens
Concept-loss weight λ			
0.3	454.89	367.84	18.89
1.0	356.71	373.40	19.71
Randomized synonyms, scrambled draw			
1	567.07	329.01	19.95
2	563.20	328.21	19.94

Table 6: **Concept-vocabulary counts under other construction settings and held-out perplexity under other training settings.** One setting changes per block. Top: the concept vocabulary $\mathcal{C}$ built from the concept-construction sample, where the adopted setting (Dice 0.15, Jaccard 1/3, complete linkage) gives the 9,061 concepts. The Dice block uses complete linkage at Jaccard 1/3, the Jaccard block Dice 0.05 with complete linkage, and the linkage block Dice 0.05 at Jaccard 1/3. Largest: words in the largest concept; $\geq$200: concepts with 200 or more words; Senses of *light*: senses found for the verb *light*. Bottom: Llama 1B after 50,000 optimizer steps (seed 0), at 5,428 concept-target positions, 82,014 content-word positions and 9,737,170 tokens (the scrambled draws at $\lambda = 0.6$).

into concepts of 200 or more words, and complete linkage does not.

Training settings. The training sweeps train Llama 1B at 2 other concept-loss weights λ and with 2 other scrambled draws of Randomized synonyms (Table 6, bottom). They are scored at 5,428 concept-target positions, not the 5,154 of Section 6.1, so they compare settings with each other. A larger concept-loss weight λ lowers concept-target perplexity and raises global perplexity, and the 2 scrambled draws give nearly equal values.

F Judged-Synonym Probability, Weighted NTP and Randomized Synonyms

Appendix F gives the details behind 3 results of Section 6: the judged-synonym probability, the alternative Weighted NTP and the content-word lead of Randomized synonyms.

Judged-synonym probability. Every held-out concept-target position has at least one contextual synonym w_j besides the observed word w_0, judged for that occurrence as for a training target, so the judged-synonym probability covers all of them. Table 7 and Fig. 7 report its arithmetic mean, which can be dominated by positions where the synonyms already hold much probability, and its geometric mean, which follows the typical position. Probabilities below 10^{-12} are raised to 10^{-12} before the logarithm, which happens only for Falcon3's base models, at 26 or fewer of their 10,028 positions. Randomized synonyms raises the geometric mean above NTP post-training's but lowers the arithmetic mean in every model.

Weighted NTP. Weighted NTP keeps the NTP and retention losses, $\mathcal{L}_{\text{NTP}}$ and $\mathcal{L}_{\text{retention}}$, and replaces the concept loss $\mathcal{L}_{\text{concept}}$ with the mean cross-entropy of the observed word w_0 over the same concept-target positions $\mathcal{S}$, at the same weight $\lambda = 0.6$. In an optimizer step, the observed word w_0 at a concept-target position thus gets weight $1/|\mathcal{I}| + \lambda/|\mathcal{S}|$ and every other position $1/|\mathcal{I}|$. Everything else is shared. The name follows the Weighted NTP control of Zhang et al. (2026), which gives content words a fixed weight (they report weight 5, whose global perplexity is closest to that of their concept-trained models). Ours reuses the concept loss's weight λ and averaging instead, which gives a concept-target position about 400 times the weight of another position in an average Llama optimizer step (5.9 targets among 3,865 positions).

Table 8 compares it with Conceptizer. Weighted NTP has the lowest concept-target perplexity of all post-trained methods in all 3 models at the smallest size. Its knowledge, sentiment and commonsense scores are comparable to Conceptizer's. However, it has a lower judged-synonym probability and a higher global perplexity than Conceptizer in all 3.

Randomized synonyms on content words. Most content-word positions hold no concept target: for Llama, 76,860 of 82,014 do not. Of these, 68,746 are the first token of a word that splits into

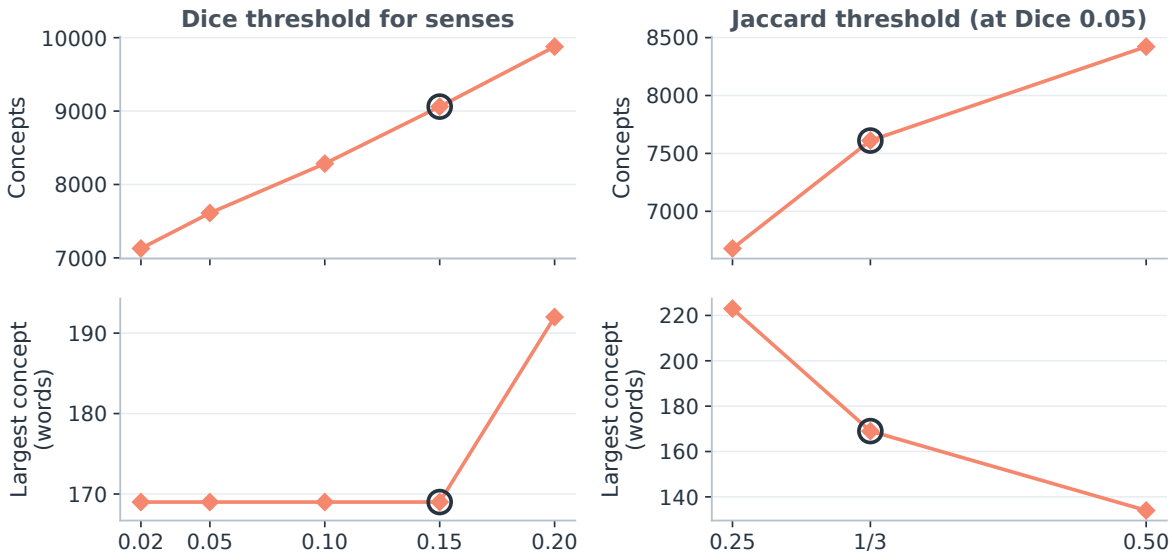


Figure 6: **A higher Dice threshold gives more concepts, and a higher Jaccard threshold gives more and smaller concepts.** The counts of Table 6 (top), one setting changed at a time; ringed: the adopted value.

	Judged-synonym probability (%) ↑							
	Base model		NTP post-training		Randomized synonyms		Conceptizer	
Model	Geo.	Arith.	Geo.	Arith.	Geo.	Arith.	Geo.	Arith.
Llama 1B	0.192	3.94	0.216	3.96	0.313	2.59	**0.464**	**5.00**
Llama 3B	0.223	4.63	0.251	4.62	0.368	2.90	**0.619**	**6.21**
Llama 8B	0.242	5.33	0.274	5.18	0.419	3.24	**0.774**	**7.38**
Qwen2.5 1.5B	0.223	4.60	0.239	4.53	0.378	2.81	**0.582**	**5.86**
Qwen2.5 3B	0.220	5.10	0.254	4.97	0.410	3.04	**0.672**	**6.50**
Qwen2.5 7B	0.207	5.51	0.261	5.39	0.434	3.35	**0.765**	**7.31**
Falcon3 1B	0.187	4.78	0.231	4.47	0.344	2.44	**0.580**	**5.50**
Falcon3 3B	0.166	5.18	0.237	4.69	0.361	2.52	**0.651**	**5.91**
Falcon3 7B	0.153	5.99	0.278	5.47	0.433	2.93	**0.862**	**7.16**

Table 7: **Conceptizer has the highest judged-synonym probability against the 3 baselines in every model, by both means.** At held-out concept-target positions at seed 0, across all model sizes. Geo., Arith.: geometric and arithmetic means over positions.

several tokens, such as *embed* in *embeddable*. Randomized synonyms' lead comes entirely from these positions: in all 9 models at seed 0, on first tokens that belong to concepts through first-token memberships it gives 1.1 to 1.4 times Conceptizer's probability, while on words that are a single token Conceptizer is higher. Scored over the whole word, the lead disappears: on 3,468 sampled multi-token words of Llama 1B's held-out text, Randomized synonyms gives its first-token gain back on the remaining tokens and matches NTP post-training (7.79 against 7.79 nats per word), while Conceptizer is 0.17 nats above both.

Our intuition is that scrambled synonyms send the reward of the concept loss $\mathcal{L}_{\text{concept}}$ to concepts unrelated to the sentence, and first-token memberships pass it on to the tokens that begin many member words: for Llama, *un-* and *re-* belong to 447 and 471 concepts, against 7 for the median member token. On sampled training batches of Llama 1B, the scrambled targets put about 3 times as much upward push on the tokens that belong to concepts only through first-token memberships (14 to 16% of the upward push on all logits, against 4 to 6% for Conceptizer's targets). Scrambling the synonyms alone reproduces this, and redrawing the closest concept $c^\star$ alone does not. These are gradients measured at the trained models, not along training. We expect the lead to disappear once concept targets also cover words that split into several tokens.

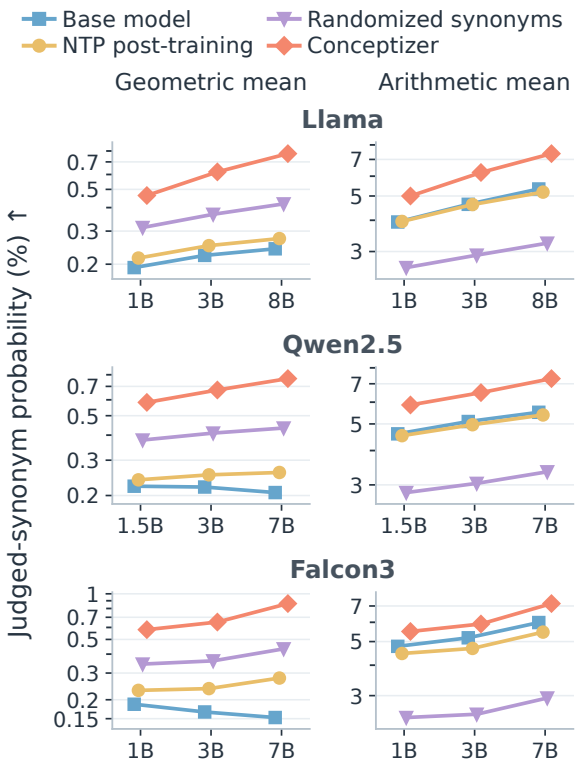


Figure 7: **Conceptizer has the highest judged-synonym probability in every model, by both means, and its probability rises with model size in every family.** Seed 0, means over held-out concept-target positions (Table 7); logarithmic axes.

Method	Held-out perplexity ↓ Concept-target	Content-word	All tokens
Llama 1B			
Weighted NTP	**218.54**	376.12	19.66
Conceptizer	367.16	**373.71**	**19.31**
Qwen2.5 1.5B			
Weighted NTP	**142.95**	235.43	14.73
Conceptizer	231.71	**232.09**	**14.48**
Falcon3 1B			
Weighted NTP	**184.93**	**403.42**	18.25
Conceptizer	306.40	410.92	**17.38**

Method	Percent ↑ Judged-synonym	Knowledge	Sentiment	Common-sense
Llama 1B				
Weighted NTP	0.298	**59.45**	56.97	68.52
Conceptizer	**0.464**	59.22	**57.13**	**68.71**
Qwen2.5 1.5B				
Weighted NTP	0.330	64.76	**55.57**	**72.95**
Conceptizer	**0.582**	**65.36**	55.48	72.31
Falcon3 1B				
Weighted NTP	0.392	**64.39**	57.50	**66.35**
Conceptizer	**0.580**	63.95	**57.54**	66.08

Table 8: **Weighted NTP has lower concept-target perplexity than Conceptizer in all 3 models, but has a lower judged-synonym probability and a higher global perplexity.** Seed 0 at the smallest model of each family. Judged-synonym: the geometric mean of Table 7; Knowledge, Sentiment and Commonsense: the mean of each group's tasks.

G MTEB

Appendix G gives the results on the 40-task MTEB suite. The MTEB suite is MTEB's English benchmark, version 2 (Muennighoff et al., 2023; Enevoldsen et al., 2025), without MindSmallReranking (Wu et al., 2020), whose data did not fit in the evaluation job's memory, so it was scored for no model. Its 40 tasks fall in 7 sections: 8 classification, 8 clustering, 3 pair classification, 1 reranking, 10 retrieval, 9 semantic textual similarity and 1 summarization, where SummEval (Fabbri et al., 2021) scores supplied summaries and nothing is generated. Every task uses the embeddings of Table 5 and its own MTEB metric. MTEB's recorded score averages the validation and test splits for 4 classification tasks (AmazonCounterfactual (O'Neill et al., 2021), MTOPDomain (Li et al., 2021), MassiveIntent and MassiveScenario (Bastianelli et al., 2020; FitzGerald et al., 2023)) and is the test score for the other 36. Table 9 and Fig. 8 give the results, and 3 of the suite's classification tasks (IMDb, ToxicConversations and TweetSentimentExtraction) also belong to the sentiment group. The other 32 tasks come from these sources: in classification, Banking77 (Casanueva et al., 2020); in clustering, StackExchange (Geigle et al., 2021), TwentyNewsgroups (Lang, 1995), and ArXivHierarchical (P2P and S2S), BiorxivP2P, MedrxivP2P, MedrxivS2S and StackExchangeP2P, which MTEB built itself; in pair classification, SprintDuplicateQuestions (Shah et al., 2018), TwitterSemEval2015 (Xu et al., 2014, 2015) and TwitterURLCorpus (Lan et al., 2017); in reranking, AskUbuntuDupQuestions (Lei et al., 2016); in retrieval, BEIR's versions (Thakur et al., 2021) of ArguAna (Wachsmuth et al., 2018), CQADupstackGaming and CQADupstackUnix (Hoogeveen et al., 2015), ClimateFEVER (Diggelmann et al., 2020), FEVER (Thorne et al., 2018), FiQA2018 (Maia et al., 2018), HotpotQA (Yang et al., 2018), SCIDOCS (Cohan et al., 2020), TRECCOVID (Wang et al., 2020; Voorhees et al., 2020) and Touche2020 (Ajjour et al., 2019; Bondarenko et al., 2020), with MTEB's hard-negative versions of ClimateFEVER, FEVER and HotpotQA and version

Model	Method	Recorded MTEB score ×100 (number of tasks) ↑							
		All (40)	Class. (8)	Clust. (8)	Pair (3)	Rerank. (1)	Retr. (10)	STS (9)	Summ. (1)
Llama 1B	Base model	**43.02**	**67.58**	41.76	**56.43**	**15.49**	**13.92**	**56.40**	14.48
	NTP post-training	42.41	66.54	**41.98**	54.88	14.96	13.63	55.46	13.26
	Rand. synonyms	41.63	66.49	41.36	52.80	14.51	12.91	53.94	14.85
	Conceptizer	42.33	67.02	41.74	54.25	15.47	13.33	55.11	**15.45**
Llama 3B	Base model	42.62	**67.79**	41.72	**58.51**	**16.23**	12.63	**56.16**	5.21
	NTP post-training	**42.73**	66.85	**42.48**	57.29	15.51	**14.15**	55.43	6.74
	Rand. synonyms	41.74	66.76	41.79	54.82	15.15	12.74	53.87	9.26
	Conceptizer	42.65	67.34	42.17	56.30	15.48	13.82	55.29	**9.82**
Llama 8B	Base model	41.87	**66.35**	43.56	**58.13**	**14.88**	11.42	54.21	4.10
	NTP post-training	42.24	66.05	**43.87**	56.98	14.55	12.94	**54.43**	5.41
	Rand. synonyms	41.10	65.50	43.19	53.32	13.53	11.79	52.56	9.92
	Conceptizer	**42.40**	66.21	43.56	56.41	14.72	**13.39**	54.39	**10.50**
Qwen2.5 1.5B	Base model	36.91	**62.58**	**40.42**	52.51	9.86	8.71	44.29	**−0.91**
	NTP post-training	**37.89**	62.54	40.37	**53.58**	**10.60**	8.69	**48.58**	−3.21
	Rand. synonyms	36.73	61.79	39.48	50.31	9.11	8.17	46.51	−1.17
	Conceptizer	37.82	62.56	40.23	52.60	9.69	**8.83**	48.48	−1.50
Qwen2.5 3B	Base model	37.13	63.42	40.46	52.46	**10.98**	8.94	44.06	−0.10
	NTP post-training	**38.38**	63.75	**40.56**	**54.10**	10.69	**9.20**	**48.48**	−0.74
	Rand. synonyms	36.73	62.98	39.56	50.51	9.82	8.05	45.24	−0.28
	Conceptizer	38.24	**63.79**	40.40	52.81	10.27	9.17	48.41	**−0.03**
Qwen2.5 7B	Base model	37.63	**62.82**	41.51	56.61	10.13	**8.53**	44.90	1.23
	NTP post-training	**38.09**	62.08	**41.56**	**56.88**	**10.57**	8.43	47.77	−1.05
	Rand. synonyms	36.26	60.70	40.35	53.39	8.79	6.67	44.96	**1.59**
	Conceptizer	37.92	61.66	41.41	56.14	9.75	8.08	**47.97**	1.49
Falcon3 1B	Base model	40.31	66.31	41.10	50.95	10.74	12.39	50.70	9.19
	NTP post-training	42.29	66.84	**42.04**	**54.40**	11.90	13.93	55.11	10.02
	Rand. synonyms	42.01	66.76	41.44	51.53	11.74	14.23	54.73	**13.74**
	Conceptizer	**42.60**	**67.11**	41.87	53.59	**11.98**	**14.33**	**55.92**	12.83
Falcon3 3B	Base model	40.81	66.60	41.90	54.33	10.30	11.84	52.58	−0.62
	NTP post-training	41.91	66.60	**42.41**	**56.07**	11.40	12.63	54.96	3.86
	Rand. synonyms	41.40	66.33	41.55	52.67	10.94	13.28	54.17	3.56
	Conceptizer	**42.28**	**66.81**	42.24	54.70	**11.71**	**13.59**	**55.65**	**6.04**
Falcon3 7B	Base model	40.79	65.15	41.29	54.64	11.21	12.74	52.11	8.57
	NTP post-training	**41.78**	65.46	**42.07**	**56.54**	**11.76**	**13.44**	**54.17**	7.82
	Rand. synonyms	40.24	64.76	40.80	52.93	10.36	12.29	51.59	8.72
	Conceptizer	41.35	**65.49**	41.50	55.45	10.86	13.37	53.10	**9.40**

Table 9: **On the whole 40-task MTEB suite, results are mixed: the best mean belongs to NTP post-training in 5 models, Conceptizer in 3 and the base model in 1.** Equal weights over the 40 tasks and within each section; means of training seeds 0, 37 and 42. Rand. synonyms: Randomized synonyms.

3 of Touche2020 (Thakur et al., 2024); and in semantic textual similarity, BIOSSES (Soğancıoğlu et al., 2017), SICK-R (Marelli et al., 2014), STS12 to STS15 (Agirre et al., 2012, 2013, 2014, 2015), STS17 and STSBenchmark (Cer et al., 2017) and STS22 (Chen et al., 2022).

H Prompts

Appendix H gives the prompts of the 3 LLMs: Llama-3.3-70B-Instruct and Qwen2.5-72B-Instruct, which propose synonym candidates, and gpt-oss-120B, the judge. Each model receives a system message, worked examples as earlier user and assistant turns, and then the input, whose slots are shown in italics. The texts are exact.

Llama-3.3-70B-Instruct, candidates in context.

System:

```
You are an English lexical-substitution annotator
producing CANDIDATE synonyms. You are given a
SENTENCE with one TARGET word marked in «double
angle brackets» and its part of speech. List the
words that could replace the TARGET *in this
sentence* with the same meaning.

This is a high-RECALL step — a separate precision
filter runs afterward — so cast a WIDE net:
include both close synonyms and looser
near-synonyms, as long as each plausibly
preserves the meaning in this context.

Rules:
- Same part of speech and grammatical form as the
target (it must drop in directly).
- SINGLE WORDS ONLY — no multi-word phrases, and
never join words together: if a fitting
replacement cannot be written as one ordinary
```

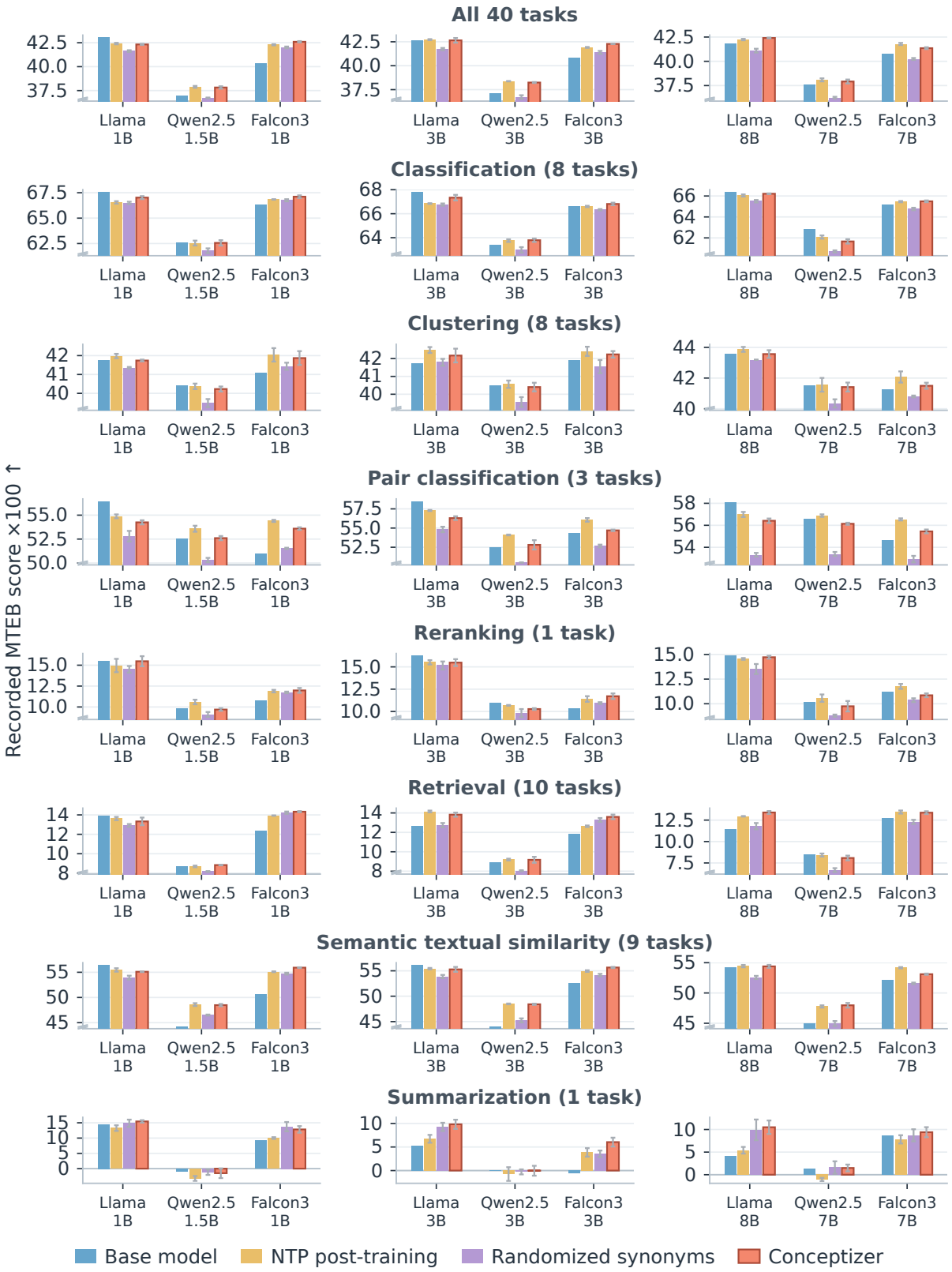


Figure 8: **By MTEB section, no method has the best mean throughout.** Means of seeds 0, 37 and 42; error bars: each method's 95% Student-t training-seed interval (values in Table 9).

word, omit it (do not output things like "celldeath" or "alkalineearthmetal").
- Never the target word itself; no duplicates; no direct antonyms.
- Err toward MORE candidates; rank closest-meaning first; up to 15.
- Output [] only if genuinely nothing could replace it.
- Output ONLY a JSON array of lowercase strings, nothing else.

User:

Sentence: The waiter carefully «placed» the warm plate on the table.
Target: "placed" (verb)

Assistant:

["set", "put", "laid", "positioned", "rested", "deposited", "planted"]

User:

Sentence: The committee will «convene» next Tuesday to vote.
Target: "convene" (verb)

Assistant:

["meet", "gather", "assemble", "congregate", "convoke"]

User:

Sentence: Marie Curie isolated «radium» in 1898.
Target: "radium" (noun)

Assistant:

[]

Input:

Sentence: *the sentence, with the word marked*
Target: *"word as written"* (*noun, verb or adjective*)

Qwen2.5-72B-Instruct, candidates without context.

System:

You are an English thesaurus. Given a WORD and its part of speech, list the single-word synonyms that could replace it in SOME context, across ALL of its meanings.

This is a comprehensive, context-free step: cover EVERY sense the word has, and include both close and looser near-synonyms. A later step selects the right ones for each context.

Rules:
- Same part of speech as the word.
- SINGLE WORDS ONLY — no multi-word phrases, and never join words together.
- Never the word itself; no duplicates; no direct antonyms.
- Cover ALL senses: if the word has several meanings, include synonyms of each; up to 25.
- Output [] only if the word has no synonyms in any sense (e.g. a proper-noun name or opaque code).
- Output ONLY a JSON array of lowercase strings, nothing else.

User:

Word: "convene" (verb)

Assistant:

["meet", "gather", "assemble", "congregate", "convoke", "summon"]

User:

Word: "bank" (noun)

Assistant:

["shore", "edge", "slope", "embankment", "treasury", "lender", "reserve", "row", "tier", "heap", "mound"]

User:

Word: "radium" (noun)

Assistant:

[]

Input:

Word: *"lemma"* (*noun, verb or adjective*)

gpt-oss-120B, the LLM judge.

System:

You are a precise English lexical-substitution judge. You are given a SENTENCE with one TARGET word marked in «double angle brackets», its part of speech, and a LIST of candidate replacement words. Do two things.

STEP 1 — QUALITY CHECK. Decide whether the TARGET is even a concept word:
- If it is a PROPER NOUN (a specific person, place, organization, or product name) or an opaque code/abbreviation, it is NOT a concept -> set "drop": true and "synonyms": [].
- A genuinely technical CONCEPT (e.g. radium, apoptosis, covalent, hypothalamus) is a real concept -> "drop": false, even if it has no good synonyms.

STEP 2 — DISCRIMINATE (only if drop is false). Keep only the candidates that could replace the TARGET *in this sentence* without changing its meaning — true in-context synonyms of the target's specific sense here.
- Same part of speech and grammatical form as the target.
- SINGLE WORDS ONLY — no multi-word phrases; never the target word itself or any phrase containing it.
- Drop wrong-sense words, antonyms, broader/narrower terms, and the target itself.
- You may ADD up to 2 obvious in-context synonyms missing from the list.
- If nothing fits, "synonyms": [] (this is fine — a real concept can have no synonyms here).

Output ONLY a JSON object: {"drop": <true|false>, "synonyms": [<lowercase words>]}. No other text.

User:

```
Sentence: She carefully «placed» the book on the
shelf.
Target: "placed" (verb)
Candidates: [set, put, laid, dropped, removed]
```

Assistant:

```
{"drop": false, "synonyms": ["set", "put",
"laid"]}
```

User:

```
Sentence: The «bank» was muddy after the heavy
flood.
Target: "bank" (noun)
Candidates: [shore, edge, slope, treasury,
lender]
```

Assistant:

```
{"drop": false, "synonyms": ["shore", "edge",
"slope"]}
```

User:

```
Sentence: The mayor met «Diana» at the new
library opening.
Target: "Diana" (noun)
Candidates: [woman, girl, name, leader]
```

Assistant:

```
{"drop": true, "synonyms": []}
```

User:

```
Sentence: Marie Curie isolated «radium» in 1898.
Target: "radium" (noun)
Candidates: [element, metal, substance, isotope]
```

Assistant:

```
{"drop": false, "synonyms": []}
```

Input:

`Sentence:` *the sentence, with the word marked*
`Target:` *"word as written"* (*noun, verb or adjective*)
`Candidates:` [*all candidates of the 3 sources, in alphabetical order, separated by commas*]

I Results for Every Training Seed

Appendix I gives every number behind the 3-seed means of Section 6. Tables 10, 11, 12 and 16 give every run's scores, and Tables 13, 14 and 15 every task's 3-seed mean. Fig. 9 plots accuracy against model size, and Fig. 10 development-set perplexity during training.

Intervals are pointwise 95% Student-t intervals over the 3 training seeds (2 degrees of freedom), on fixed evaluation items. Each evaluation job scores the post-trained methods of a seed together on one H100 GPU. The paired intervals of Section 6 come from these per-seed scores: differences are paired within seed, and a difference from the base model subtracts its single run from each seed's score. All Randomized synonyms runs share a single scrambled draw (2 other draws in Appendix E).

	Accuracy (%) ↑				
Method	Seed 0	37	42	Mean	95% CI
Llama 1B					
Base model	one fixed run			57.79	
NTP post-training	58.01	58.49	57.99	58.16	±0.70
Rand. synonyms	58.62	58.89	58.90	58.80	±0.40
Conceptizer	59.22	59.33	59.57	**59.38**	±0.44
Llama 3B					
Base model	one fixed run			62.44	
NTP post-training	62.96	63.29	63.07	63.11	±0.42
Rand. synonyms	63.21	63.52	62.75	63.16	±0.97
Conceptizer	64.04	63.96	63.86	**63.95**	±0.23
Llama 8B					
Base model	one fixed run			**68.96**	
NTP post-training	67.76	67.98	68.34	68.03	±0.73
Rand. synonyms	67.62	67.67	68.11	67.80	±0.67
Conceptizer	68.98	67.87	68.85	68.57	±1.51
Qwen2.5 1.5B					
Base model	one fixed run			64.16	
NTP post-training	64.01	64.05	63.82	63.96	±0.31
Rand. synonyms	64.48	64.13	64.25	64.28	±0.44
Conceptizer	65.36	65.17	65.06	**65.20**	±0.38
Qwen2.5 3B					
Base model	one fixed run			63.44	
NTP post-training	65.63	65.56	66.19	65.80	±0.86
Rand. synonyms	66.96	66.51	66.62	66.70	±0.58
Conceptizer	67.51	67.69	67.89	**67.69**	±0.47
Qwen2.5 7B					
Base model	one fixed run			69.66	
NTP post-training	70.54	70.81	70.98	70.78	±0.55
Rand. synonyms	70.93	70.82	71.18	70.98	±0.47
Conceptizer	71.44	71.55	71.53	**71.51**	±0.14
Falcon3 1B					
Base model	one fixed run			61.76	
NTP post-training	62.26	62.17	62.39	62.27	±0.28
Rand. synonyms	62.81	62.69	62.89	62.80	±0.25
Conceptizer	63.95	64.05	64.20	**64.06**	±0.31
Falcon3 3B					
Base model	one fixed run			66.73	
NTP post-training	65.60	65.78	65.83	65.74	±0.31
Rand. synonyms	65.42	65.64	65.44	65.50	±0.31
Conceptizer	66.88	66.89	66.63	**66.80**	±0.36
Falcon3 7B					
Base model	one fixed run			69.50	
NTP post-training	69.42	69.25	69.23	69.30	±0.26
Rand. synonyms	69.10	69.25	69.08	69.14	±0.24
Conceptizer	70.28	70.32	70.57	**70.39**	±0.39

Table 10: **Knowledge group at every training seed: Conceptizer has the best mean against the 3 baselines in 8 of 9 models.** The mean of ARC-Challenge, ARC-Easy, OpenBookQA and SciQ; 95% CI: margin of error of the Student-t training-seed interval. Rand. synonyms: Randomized synonyms.

Method	Test-split accuracy (%) ↑ Seed 0	37	42	Mean	95% CI
Llama 1B					
Base model	one fixed run			56.89	
NTP post-training	56.19	56.37	56.38	56.31	±0.26
Rand. synonyms	55.90	56.17	55.96	56.01	±0.36
Conceptizer	57.13	57.20	57.27	**57.20**	±0.17
Llama 3B					
Base model	one fixed run			56.96	
NTP post-training	56.73	56.68	56.66	56.69	±0.09
Rand. synonyms	56.30	56.39	56.32	56.34	±0.12
Conceptizer	57.60	57.79	57.70	**57.70**	±0.24
Llama 8B					
Base model	one fixed run			56.95	
NTP post-training	56.98	56.94	56.84	56.92	±0.18
Rand. synonyms	56.06	56.44	56.34	56.28	±0.49
Conceptizer	57.72	57.82	57.75	**57.76**	±0.13
Qwen2.5 1.5B					
Base model	one fixed run			55.33	
NTP post-training	54.86	54.98	54.98	54.94	±0.17
Rand. synonyms	54.64	54.57	54.64	54.61	±0.09
Conceptizer	55.48	55.37	55.35	**55.40**	±0.17
Qwen2.5 3B					
Base model	one fixed run			55.87	
NTP post-training	55.43	55.47	55.51	55.47	±0.10
Rand. synonyms	54.69	55.00	54.86	54.85	±0.38
Conceptizer	56.10	56.00	56.26	**56.12**	±0.32
Qwen2.5 7B					
Base model	one fixed run			55.94	
NTP post-training	56.15	56.25	56.22	56.21	±0.12
Rand. synonyms	55.52	55.53	55.47	55.51	±0.08
Conceptizer	56.82	56.72	56.75	**56.76**	±0.12
Falcon3 1B					
Base model	one fixed run			57.12	
NTP post-training	56.61	56.64	56.67	56.64	±0.07
Rand. synonyms	56.74	56.51	56.63	56.63	±0.29
Conceptizer	57.54	57.59	57.48	**57.54**	±0.13
Falcon3 3B					
Base model	one fixed run			**58.14**	
NTP post-training	57.06	57.07	57.04	57.06	±0.03
Rand. synonyms	56.41	56.39	56.57	56.46	±0.25
Conceptizer	57.95	57.76	57.86	57.86	±0.24
Falcon3 7B					
Base model	one fixed run			57.58	
NTP post-training	57.40	57.48	57.48	57.45	±0.11
Rand. synonyms	56.49	56.54	56.54	56.52	±0.08
Conceptizer	58.17	58.31	58.31	**58.26**	±0.20

Table 11: **Sentiment group at every training seed: Conceptizer has the best mean against the 3 baselines in 8 of 9 models.** The mean of IMDb, ToxicConversations, TweetSentimentExtraction, Emotion, ToxicChat, PoemSentiment, FrenkEn, YelpReviewFull and AmazonPolarity; 95% CI: margin of error of the Student-t training-seed interval. Rand. synonyms: Randomized synonyms.

Method	Accuracy (%) ↑ Seed 0	37	42	Mean	95% CI
Llama 1B					
Base model	one fixed run			67.02	
NTP post-training	66.72	67.15	66.87	66.91	±0.54
Rand. synonyms	67.80	67.43	67.39	67.54	±0.55
Conceptizer	68.71	68.18	69.20	**68.69**	±1.27
Llama 3B					
Base model	one fixed run			74.16	
NTP post-training	75.81	75.30	75.09	75.40	±0.92
Rand. synonyms	75.32	74.23	74.89	74.81	±1.36
Conceptizer	75.78	75.53	75.16	**75.49**	±0.78
Llama 8B					
Base model	one fixed run			**82.41**	
NTP post-training	82.04	82.25	81.81	82.03	±0.55
Rand. synonyms	81.15	80.57	81.23	80.98	±0.89
Conceptizer	82.28	81.05	81.61	81.64	±1.53
Qwen2.5 1.5B					
Base model	one fixed run			71.11	
NTP post-training	72.10	72.60	73.04	**72.58**	±1.17
Rand. synonyms	71.18	71.49	71.50	71.39	±0.46
Conceptizer	72.31	72.40	72.14	72.28	±0.32
Qwen2.5 3B					
Base model	one fixed run			76.44	
NTP post-training	77.10	77.47	76.67	**77.08**	±1.00
Rand. synonyms	75.35	74.92	74.56	74.94	±0.98
Conceptizer	76.14	76.97	76.99	76.70	±1.21
Qwen2.5 7B					
Base model	one fixed run			80.13	
NTP post-training	81.89	82.21	81.79	**81.97**	±0.55
Rand. synonyms	80.62	80.71	80.46	80.60	±0.32
Conceptizer	81.73	82.05	81.85	81.88	±0.40
Falcon3 1B					
Base model	one fixed run			65.79	
NTP post-training	65.94	65.84	65.95	65.91	±0.15
Rand. synonyms	65.19	65.23	65.09	65.17	±0.18
Conceptizer	66.08	66.30	66.46	**66.28**	±0.47
Falcon3 3B					
Base model	one fixed run			**69.90**	
NTP post-training	69.03	69.32	69.19	69.18	±0.36
Rand. synonyms	68.08	67.98	67.31	67.79	±1.04
Conceptizer	68.51	67.88	67.91	68.10	±0.88
Falcon3 7B					
Base model	one fixed run			**78.56**	
NTP post-training	78.04	78.52	78.44	78.33	±0.64
Rand. synonyms	77.07	76.71	76.81	76.86	±0.46
Conceptizer	77.75	77.71	77.61	77.69	±0.17

Table 12: **Commonsense group at every training seed: the best mean belongs to Conceptizer, NTP post-training and the base model in 3 models each.** The mean of HellaSwag, COPA and WinoGrande; 95% CI: margin of error of the Student-t training-seed interval. Rand. synonyms: Randomized synonyms.

Method	Knowledge accuracy (%) ↑			
	ARC-C	ARC-E	OBQA	SciQ
Llama 1B				
Base model	37.71	63.64	37.60	92.20
NTP post-training	37.91	64.91	38.13	91.70
Randomized synonyms	38.40	65.52	39.07	92.23
Conceptizer	**38.68**	**66.02**	**40.13**	**92.67**
Llama 3B				
Base model	45.99	71.38	39.20	93.20
NTP post-training	45.36	71.80	40.87	**94.40**
Randomized synonyms	**46.05**	70.93	41.47	94.20
Conceptizer	45.96	**72.31**	**43.20**	94.33
Llama 8B				
Base model	**55.29**	**79.76**	45.00	95.80
NTP post-training	52.84	79.17	44.47	95.63
Randomized synonyms	53.67	78.10	43.87	95.57
Conceptizer	54.38	78.55	**45.07**	**96.27**
Qwen2.5 1.5B				
Base model	46.50	75.84	40.20	**94.10**
NTP post-training	46.64	75.36	40.73	93.10
Randomized synonyms	46.22	75.35	42.13	93.43
Conceptizer	**48.27**	**76.84**	**42.53**	93.17
Qwen2.5 3B				
Base model	48.04	72.81	41.60	91.30
NTP post-training	49.15	76.50	42.27	95.27
Randomized synonyms	49.69	76.81	45.00	95.30
Conceptizer	**51.22**	**78.75**	**45.47**	**95.33**
Qwen2.5 7B				
Base model	55.29	80.93	48.80	93.60
NTP post-training	57.25	82.15	48.67	95.03
Randomized synonyms	57.37	81.97	49.20	**95.37**
Conceptizer	**58.67**	**82.95**	**49.33**	95.07
Falcon3 1B				
Base model	45.31	68.01	40.60	93.10
NTP post-training	44.65	69.77	40.60	94.07
Randomized synonyms	44.48	70.83	41.53	**94.33**
Conceptizer	**46.79**	**72.00**	**43.20**	94.27
Falcon3 3B				
Base model	**54.27**	74.87	42.40	**95.40**
NTP post-training	49.72	**76.23**	41.87	95.13
Randomized synonyms	50.63	74.45	42.47	94.47
Conceptizer	52.47	75.90	**43.87**	94.97
Falcon3 7B				
Base model	59.73	77.86	45.60	**94.80**
NTP post-training	58.48	**79.59**	44.47	94.67
Randomized synonyms	57.88	78.40	45.73	94.57
Conceptizer	**59.95**	79.41	**48.07**	94.13

Table 13: **Conceptizer has the best knowledge accuracy on OpenBookQA in all 9 models and on ARC-Challenge and ARC-Easy in 6 each.** Means of training seeds 0, 37 and 42.

Method	Commonsense accuracy (%) ↑		
	HellaSwag	COPA	WinoGrande
Llama 1B			
Base model	61.57	78.00	61.48
NTP post-training	**61.96**	77.00	61.77
Randomized synonyms	61.12	80.33	61.17
Conceptizer	61.74	**82.33**	**62.01**
Llama 3B			
Base model	71.65	82.00	68.82
NTP post-training	**71.93**	84.33	**69.93**
Randomized synonyms	71.55	84.33	68.56
Conceptizer	71.85	**85.67**	68.96
Llama 8B			
Base model	**79.59**	**94.00**	73.64
NTP post-training	79.28	93.00	73.82
Randomized synonyms	78.53	91.33	73.09
Conceptizer	78.75	92.33	**73.85**
Qwen2.5 1.5B			
Base model	**68.26**	82.00	63.06
NTP post-training	67.39	**86.00**	64.35
Randomized synonyms	66.99	82.67	64.51
Conceptizer	67.38	84.33	**65.14**
Qwen2.5 3B			
Base model	**74.96**	85.00	69.38
NTP post-training	73.61	**87.33**	70.30
Randomized synonyms	72.87	81.67	70.30
Conceptizer	73.45	85.67	**70.98**
Qwen2.5 7B			
Base model	**80.44**	89.00	70.96
NTP post-training	79.08	92.33	**74.48**
Randomized synonyms	78.18	90.00	73.61
Conceptizer	78.80	**92.67**	74.16
Falcon3 1B			
Base model	**63.14**	74.00	60.22
NTP post-training	62.38	75.00	**60.35**
Randomized synonyms	61.57	75.00	58.93
Conceptizer	62.66	**76.67**	59.51
Falcon3 3B			
Base model	**70.15**	**75.00**	64.56
NTP post-training	67.89	74.67	**64.98**
Randomized synonyms	66.35	74.33	62.69
Conceptizer	67.18	73.67	63.46
Falcon3 7B			
Base model	**78.43**	**87.00**	70.24
NTP post-training	76.48	86.67	**71.85**
Randomized synonyms	75.20	85.33	70.06
Conceptizer	75.54	86.00	71.53

Table 14: **The base model has the best HellaSwag accuracy in 7 of 9 models, Conceptizer the best COPA accuracy in 4, and NTP post-training the best WinoGrande accuracy in 5 (Conceptizer in the other 4).** Means of training seeds 0, 37 and 42.

Method	Test-split accuracy (%) ↑								
	IMDb	ToxConv.	TweetSent.	Emotion	ToxicChat	Poem	FrenkEn	Yelp	Amazon
Llama 1B									
Base model	**68.40**	65.47	51.88	**33.28**	67.63	42.31	58.72	**51.76**	72.52
NTP post-training	67.27	65.16	51.86	32.14	66.38	42.44	58.67	51.30	71.59
Randomized synonyms	66.36	65.92	51.82	32.17	66.04	41.79	58.19	51.26	70.53
Conceptizer	68.04	**66.38**	**52.22**	33.21	**68.57**	**43.33**	**59.09**	51.21	**72.74**
Llama 3B									
Base model	69.48	65.21	51.95	**32.86**	67.68	40.67	59.25	**53.32**	72.25
NTP post-training	68.89	65.07	51.93	31.57	67.36	40.96	59.30	53.01	72.12
Randomized synonyms	67.37	65.51	51.79	31.75	67.50	40.83	59.07	52.94	70.28
Conceptizer	**70.01**	**66.05**	**52.28**	32.51	**69.63**	**42.85**	**60.14**	52.44	**73.36**
Llama 8B									
Base model	70.24	63.67	52.68	31.53	68.10	38.65	59.21	**54.30**	74.15
NTP post-training	69.85	64.53	52.58	31.51	66.63	39.04	60.11	53.97	74.06
Randomized synonyms	67.18	64.34	52.19	31.29	66.77	38.78	59.73	54.19	72.02
Conceptizer	**71.08**	**64.68**	**52.79**	**32.10**	**68.65**	**40.64**	**60.23**	54.09	**75.59**
Qwen2.5 1.5B									
Base model	**68.61**	62.08	51.57	29.76	**64.50**	43.08	55.77	52.35	**70.23**
NTP post-training	67.72	62.18	51.73	30.20	62.97	42.88	**55.81**	51.66	69.31
Randomized synonyms	66.65	61.86	51.26	29.66	62.65	43.53	55.41	**52.37**	68.14
Conceptizer	68.03	**62.78**	**51.95**	**30.72**	63.64	**44.01**	55.68	51.79	69.99
Qwen2.5 3B									
Base model	69.42	64.16	52.08	29.86	64.93	**41.35**	56.75	52.41	**71.85**
NTP post-training	68.54	63.95	51.90	29.40	64.99	40.06	56.80	52.57	71.00
Randomized synonyms	67.01	63.89	51.25	29.57	64.94	38.01	56.77	**53.16**	69.02
Conceptizer	**69.55**	**64.81**	**52.17**	**30.19**	**66.19**	40.61	**57.42**	52.66	71.46
Qwen2.5 7B									
Base model	**71.03**	**63.88**	50.81	29.28	68.15	38.08	57.57	52.17	72.48
NTP post-training	70.04	62.81	50.96	29.82	66.96	42.15	58.14	52.62	72.36
Randomized synonyms	66.93	63.17	50.47	29.77	67.30	41.09	57.41	**53.24**	70.20
Conceptizer	70.22	63.56	**51.03**	**30.56**	**68.28**	**43.40**	**58.31**	52.39	**73.09**
Falcon3 1B									
Base model	70.12	66.01	51.79	31.08	69.64	**42.98**	**57.63**	52.00	72.81
NTP post-training	69.10	65.84	52.54	31.44	68.91	41.89	56.19	51.44	72.41
Randomized synonyms	68.30	66.09	52.15	31.31	69.81	42.69	56.08	**52.13**	71.08
Conceptizer	**70.46**	**66.34**	**52.73**	**32.59**	**70.58**	42.63	56.94	51.51	**74.04**
Falcon3 3B									
Base model	**71.25**	**65.61**	51.81	31.74	**73.45**	**43.17**	**59.24**	51.90	75.13
NTP post-training	69.98	64.77	52.78	31.82	68.02	41.79	58.41	51.64	74.32
Randomized synonyms	68.23	64.40	52.24	31.81	68.41	41.19	57.59	**52.40**	71.85
Conceptizer	**71.25**	64.88	**53.03**	**33.13**	69.46	42.88	58.67	51.94	**75.48**
Falcon3 7B									
Base model	**72.95**	65.92	51.75	30.57	**68.78**	40.38	59.19	**52.96**	75.70
NTP post-training	71.94	65.05	52.33	30.77	67.54	43.04	58.72	52.30	75.39
Randomized synonyms	68.57	65.34	51.67	30.46	66.84	42.05	58.77	52.47	72.54
Conceptizer	72.77	**65.97**	**52.87**	**31.77**	67.92	**44.68**	**59.33**	52.53	**76.53**

Table 15: **Conceptizer has the best sentiment accuracy on TweetSentimentExtraction in all 9 models, on the 7 other tasks in 5 to 7 each, and on YelpReviewFull in none.** Means of training seeds 0, 37 and 42.

	Held-out perplexity ↓								
	Concept-target			Content-word			All tokens		
Method	Seed 0	37	42	Seed 0	37	42	Seed 0	37	42
Llama 1B									
Base model		1229.58			500.78			21.12	
NTP post-training	1014.70	1012.41	1018.06	384.95	384.51	384.94	17.84	17.85	17.85
Rand. synonyms	548.33	551.42	551.19	331.06	331.49	331.61	19.88	19.88	19.88
Conceptizer	367.16	370.37	369.72	373.71	373.73	374.16	19.31	19.32	19.31
Llama 3B									
Base model		760.82			297.02			17.08	
NTP post-training	616.36	617.81	617.13	225.41	225.85	225.45	14.47	14.47	14.47
Rand. synonyms	339.40	339.76	339.68	203.98	203.80	203.84	16.18	16.18	16.18
Conceptizer	218.14	218.27	218.44	225.52	225.84	225.40	15.68	15.68	15.68
Llama 8B									
Base model		391.41			145.11			13.11	
NTP post-training	338.68	340.66	338.46	118.15	117.97	118.11	11.39	11.39	11.39
Rand. synonyms	199.04	200.17	198.09	111.67	111.45	111.51	12.62	12.62	12.62
Conceptizer	128.97	128.85	128.12	119.56	119.67	119.40	12.19	12.20	12.20
Qwen2.5 1.5B									
Base model		723.06			275.97			14.28	
NTP post-training	663.10	664.37	663.30	247.50	247.80	247.76	13.73	13.73	13.73
Rand. synonyms	361.35	362.86	362.12	206.21	206.14	206.01	15.07	15.07	15.07
Conceptizer	231.71	231.22	230.95	232.09	232.25	232.02	14.48	14.48	14.48
Qwen2.5 3B									
Base model		551.47			198.26			12.69	
NTP post-training	478.69	480.38	479.81	174.88	174.91	174.84	12.16	12.16	12.16
Rand. synonyms	269.38	267.79	268.43	150.56	150.58	150.69	13.34	13.33	13.33
Conceptizer	167.77	167.39	167.27	165.61	165.48	165.61	12.81	12.80	12.81
Qwen2.5 7B									
Base model		435.07			145.12			11.33	
NTP post-training	358.80	359.11	359.39	124.32	124.35	124.38	10.77	10.77	10.77
Rand. synonyms	199.65	200.63	200.50	112.08	111.98	112.20	11.77	11.77	11.78
Conceptizer	128.09	128.63	128.22	120.86	120.59	120.52	11.34	11.34	11.34
Falcon3 1B									
Base model		1377.40			651.94			18.94	
NTP post-training	1127.16	1128.16	1127.95	501.12	500.89	500.74	16.04	16.04	16.04
Rand. synonyms	639.96	639.27	639.39	393.09	393.26	393.34	18.21	18.21	18.21
Conceptizer	306.40	307.15	306.83	410.92	410.82	410.99	17.38	17.38	17.37
Falcon3 3B									
Base model		1259.24			554.76			17.59	
NTP post-training	917.72	918.48	917.56	389.43	389.54	389.61	14.80	14.80	14.80
Rand. synonyms	509.78	509.63	510.11	318.23	318.24	318.25	16.79	16.79	16.79
Conceptizer	242.86	243.42	242.87	327.66	327.45	327.40	16.06	16.06	16.06
Falcon3 7B									
Base model		980.95			314.38			15.09	
NTP post-training	555.47	554.70	555.21	193.90	193.86	193.97	12.03	12.03	12.03
Rand. synonyms	334.06	333.99	334.84	175.48	175.68	175.89	13.67	13.67	13.67
Conceptizer	151.04	151.28	150.84	170.36	170.56	170.37	13.05	13.06	13.06

Table 16: **Held-out perplexity at every training seed: Conceptizer has the lowest perplexity against the 3 baselines at concept-target positions, and NTP post-training the lowest over all tokens.** 3-seed means in Fig. 4. Rand. synonyms: Randomized synonyms.

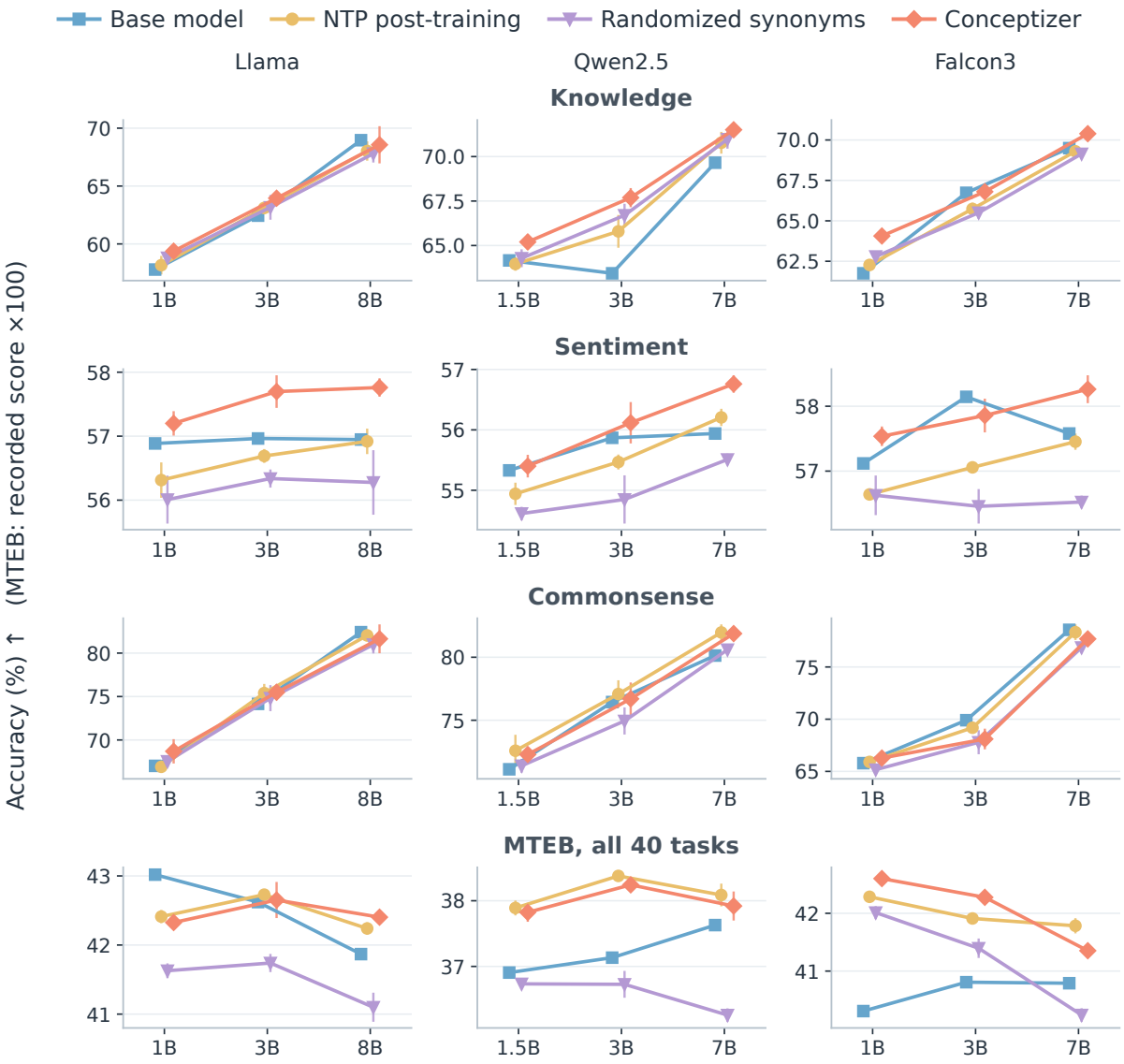


Figure 9: **Downstream performance against model size, by family.** Means of training seeds 0, 37 and 42 with 95% Student-t intervals (Tables 10, 11, 12 and 9).

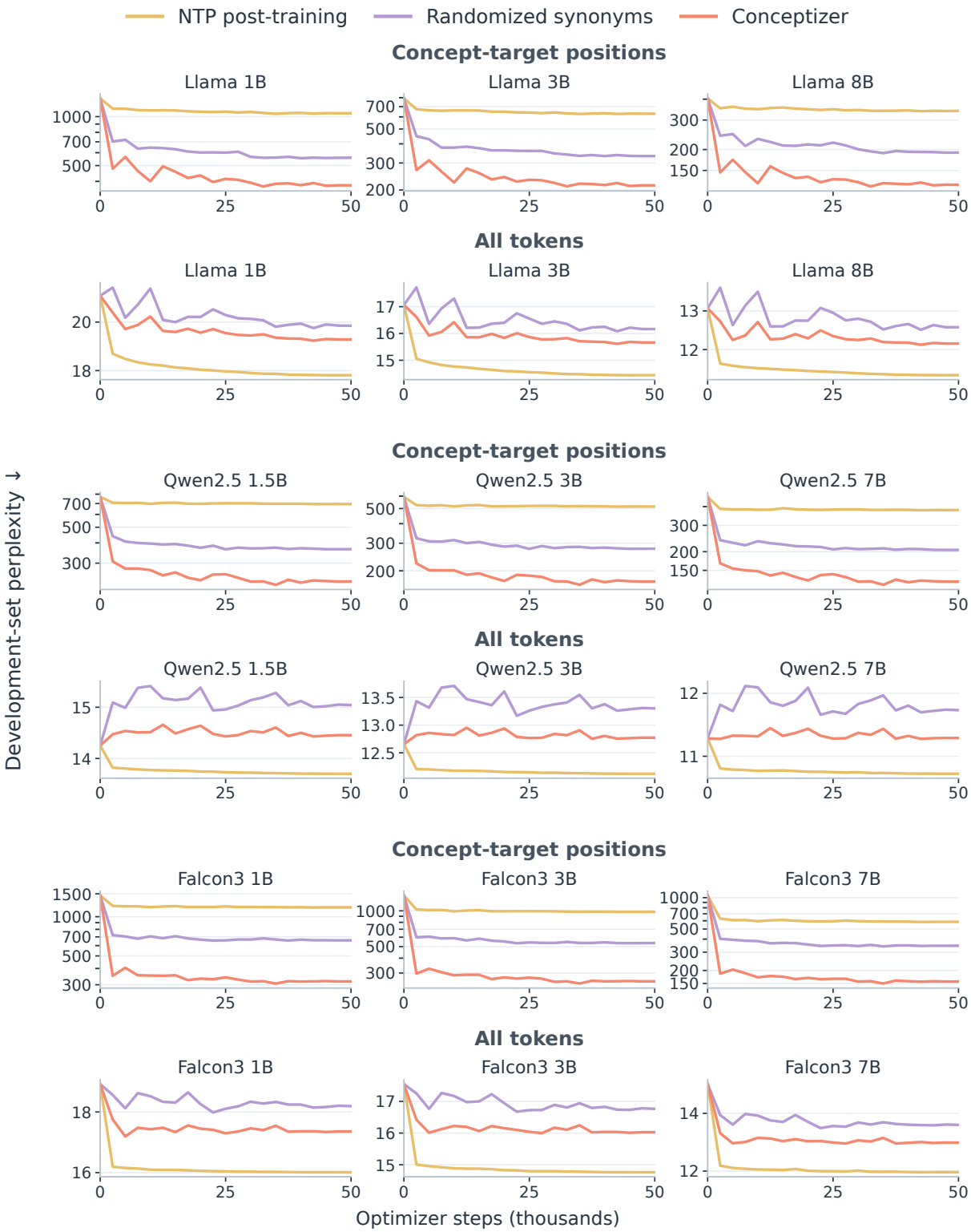


Figure 10: **Development-set perplexity during training.** Scored every 2,500 optimizer steps; step 0 is the base model. Lines: means of training seeds 0, 37 and 42; concept-target rows on logarithmic axes.