DisCoCLIP: A Distributional Compositional Tensor Network Encoder for Vision-Language Understanding

Kin Ian Lo and **Hala Hawashin** and **Mina Abbaszadeh**
Tilen Limback-Stokin and **Hadi Wazni** and **Mehrnoosh Sadrzadeh**
University College London

Abstract

Recent vision–language models excel at large-scale image–text alignment but often neglect the compositional structure of language, leading to failures on tasks that hinge on word order and predicate–argument structure. We introduce DisCoCLIP, a multimodal encoder that combines a frozen CLIP vision transformer with a novel tensor network text encoder that explicitly encodes syntactic structure. Sentences are parsed with a Combinatory Categorial Grammar parser to yield distributional word tensors whose contractions mirror the sentence's grammatical derivation. To keep the model efficient, high-order tensors are factorized with tensor decompositions, reducing parameter count from tens of millions to under one million. Trained end-to-end with a self-supervised contrastive loss, DisCoCLIP markedly improves sensitivity to verb semantics and word order: it raises CLIP's SVO-Probes verb accuracy from 77.6% to 82.4%, boosts ARO attribution and relation scores by over 9% and 4%, and achieves 93.7% on a newly introduced SVO-Swap benchmark. These results demonstrate that embedding explicit linguistic structure via tensor networks yields interpretable, parameter-efficient representations that substantially improve compositional reasoning in vision–language tasks.

1 Introduction

Vision-language understanding is a key challenge in AI, with applications to image captioning and multimodal retrieval. Models like OpenAI's CLIP (Radford et al., 2021) have shown that large-scale joint embeddings can effectively connect visual and textual data. However, these models mainly rely on Transformer architectures with dense attention, which may overlook the linguistic structure. For instance, recent evaluations of CLIP-like models show that they often ignore word order, acting like bags-of-words (Thrush et al., 2022; Jiang et al., 2024; Li et al., 2024). The Attribution, Relation and Order (ARO) benchmark (Yuksekgonul et al., 2023) checks if they are able to understand the correct word order. Similarly, the SVO-probes benchmark (Hendricks and Nematzadeh, 2021) tests if these models mainly focus on nouns, or are also able to recognise verbs. Both of these issues have been common challenges for vision-language models.

It has been argued that these challenges stem from CLIP-like models being trained on web-sourced image-caption pairs, where captions (often alt-texts) frequently ignore word order and verb usage. As a result, their contrastive learning is not sensitive to linguistic structure (Yuksekgonul et al., 2023). While training with *hard negatives* could address this, such samples are costly to source. Instead, we introduce **DisCoCLIP**, the first model for vision and language with a text encoder that fully incorporates the compositional linguistic structure of text with the distributions of the words therein. To achieve this, we represent sentences as tensor networks, where each word is encoded as a tensor and interactions between words are captured through a series of tensor contractions.

The advantages of using a tensor network text encoder are twofold. First, it enables explicit encoding of both syntactic structure and statistical semantic information, making the resulting text representations more interpretable than those produced by transformer-based encoders. Second, tensor network decompositions can dramatically reduce the number of parameters required, allowing for efficient modelling of high-order interactions without incurring exponential growth in tensor size. Tensor networks are widely used in quantum machine learning to capture higher order data correlations (Biamonte et al., 2017; Schuld et al., 2015; Stoudenmire and Schwab, 2016; Cichocki et al., 2016). Their use in vision-language tasks might lead to further advantages coming from the quan-

*Proceedings of the 14th Joint Conference on Lexical and Computational Semantics (*SEM 2025)*, pages 316–327
November 8-9, 2025 ©2025 Association for Computational Linguistics

tum world.

DisCoCLIP was evaluated on two existing benchmarks on compositional capability: SVO-Probes and ARO, as well as on a new SVO-Swap benchmark created by swapping subjects and objects. We compare the performance of **DisCoCLIP** with CLIP, OpenCLIP (Ilharco et al., 2021) and BLIP (Li et al., 2022) on these benchmarks.

DisCoCLIP outperforms CLIP and OpenCLIP on verb understanding by 4.82% and 1.01%. It also outperforms CLIP in overall performance by 1.3%, but falls behind OpenCLIP and BLIP by 2.05% and 7.9%. On SVO-Swap, it achieves an accuracy of 93.68% outperforming all three of CLIP, OpenCLIP and BLIP by a large margin $(30.52\% - 57.04\%)$. On ARO-Relation, again it outperforms all three of the CLIP models by $4.28\% - 5.1\%$, in ARO-Attribution, it outperforms CLIP and OpenCLIP by 9.01% and 10.88%, but falls behind BLIP by 8.45%.

In summary, DisCoCLIP achieves comparable performance to transformer-based models with orders of magnitude fewer parameters. The use of tensor decomposition enables efficient representation and computation, making our model more parameter-efficient and potentially more robust when training data is limited. To our knowledge, it is the first time that the theory of tensor networks has been used to model the structure of language or used in vision-language tasks. Our work provides a new witness for the applications of tensor networks to machine learning and further showcases the advantage of using them.

2 Related Work

Several approaches have been proposed to address these challenges in vision-language models. Some incorporate aspects of linguistic structure (Jiang et al., 2024), others introduce hard negatives (Li et al., 2024), and some incentivize learning by explicitly rewarding the model for capturing linguistic elements such as adjectives and verbs (Thrush et al., 2022).

Tensor networks were introduced to make the numerical treatment of many-body quantum states feasible by exploiting their internal structure (White, 1992). Such states naturally live in exponentially large tensor-product spaces, which are difficult to handle directly. A tensor network circumvents this by factorizing a single, high-order tensor into a set of lower-order tensors, whose indices are glued together by contraction operations. In a *Tensor Train* (also known as a Matrix Product State, or MPS), these tensors are arranged in a strictly one-dimensional sequence, with each tensor contracted only to its immediate predecessor and successor through shared *bond* indices; by contrast, a Tree Tensor Network connects tensors in a branching, hierarchical structure. Tensor networks have found applications outside physics, especially in machine learning where they are used for sequence modelling (Harvey et al., 2025), optimizing the computations of neural networks (Ahromi and Orús, 2024; Novikov et al., 2015), and in general any large-scale optimization problem (Cichocki et al., 2017), such as latent feature extraction (Stoudenmire, 2018) and security (Aizpurua et al., 2025). Their decomposition methods have been tested on image classification tasks (Roberts et al., 2019; Rao et al., 2020; Serafini and d'Avila Garcez, 2017), word statistics, and document retrieval from large corpora of text (Miller et al., 2021; Zhang et al., 2019; Liu et al., 2005; Bouchard et al., 2015).

Tensors and the contraction operation between them were also used in a model of meaning known as "compositional distributional semantics" (Baroni and Zamparelli, 2010; Maillard et al., 2014; Grefenstette and Sadrzadeh, 2011; Yeung and Kartsaklis, 2021). In this model, the meaning of each word is either a vector or a higher-order tensor. The orders of the tensors are determined by the grammatical roles of words. Meanings of nouns are vectors, where as meanings of words with functional roles such as adjectives and verbs are matrices and cubes. **DisCoCLIP** is inspired by compositional distributional semantics and the theory of tensor networks. We denote the meaning of a piece of text by a tensor network. In this tensor network, the tensors encode meanings of words, the layout of the tensor network represents the syntactic structure of the sentence. Other tensor network layouts are used as baselines to test how useful is encoding less structure, such as word order and bags-of-words.

Another key novelty of our model is that it extends compositional distributional semantics to a multimodal setting. Previous multimodal adaptations include (Lewis et al., 2024) for compositional concept learning, (Nazir and Sadrzadeh, 2024) for audio-text retrieval, and (Wazni et al., 2024) for verb understanding in CLIP. However, **DisCoCLIP** differs from these approaches in two important ways. First, our model is more general: It handles sentences of arbitrary syntactic structure,

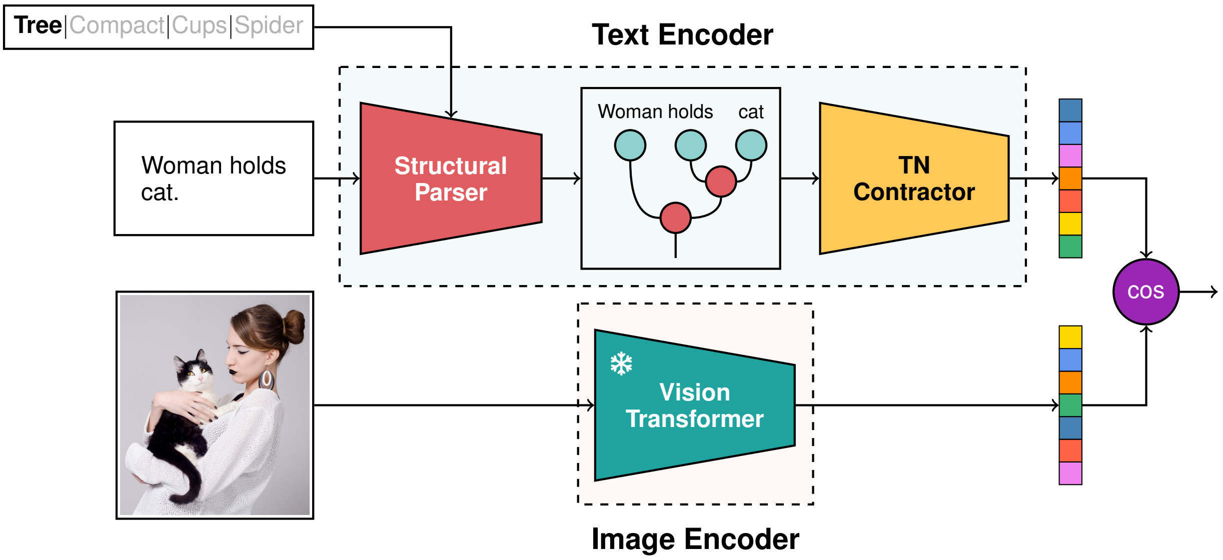


Figure 1: An illustration of the architecture of DisCoCLIP, which consists of a text encoder based on a structure-informed tensor network of words, and a vision encoder based on a Vison Transformer (ViT). The Structural Parser converts the input text into a tensor network, based on the chosen structure which could be any of the four types: **Tree**, **Compact**, **Cups** or **Spider**. The tensor network is then contracted by the Tensor Network Contractor, which computes an optimal contracting order to obtain a single vector representing the meaning of the input text. The input image is processed by Vision Transformer (ViT) to obtain a vector representation of the image. The text and image vectors are then used to compute a similarity score, which is used for training the model and for downstream evaluation.

whereas prior work typically focuses on specific constructions such as subject-verb-object (Lewis et al., 2024; Wazni et al., 2024) or adjective-noun pairs (Nazir and Sadrzadeh, 2024). Second, **DisCoCLIP** features an end-to-end pipeline trained with a single objective function, in contrast to previous methods that require a separate objective for their different text and audio/image model components. This unified approach enables more flexible and scalable multimodal learning.

3 Basics of Tensor Networks

A tensor network is a collection of tensors contracted together to form a new tensor. An order-n tensor T is a multi-dimensional array $T \in \mathbb{R}^{d_1 \times \cdots \times d_n}$, where d_i is the dimension of the i-th index. Elements are denoted by $T_{i_1,\ldots,i_n}$, with each i_k ranging from 0 to d_{k-1}. Scalars, vectors, and matrices are tensors of order 0, 1, and 2, respectively.

Tensor contractions. Tensors can be multiplied together by contracting over a shared index, which generalizes matrix multiplication. For example, given two tensors $A \in \mathbb{R}^{d_1 \times d_2}$ and $B \in \mathbb{R}^{d_2 \times d_3}$, their contraction over the second index yields a new tensor $C \in \mathbb{R}^{d_1 \times d_3}$:

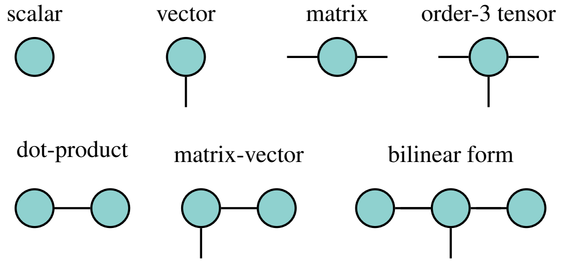


Figure 2: Graphical representation of tensor networks. A tensor is depicted as a node with one edge for each index of the tensor. For example a scalar has no edge, a vector has one edge, a matrix has two edges and an order-3 tensor has 3 edges. An edge of a node can be connected to another edge of another node, forming a *contraction*, which is a generalised form of matrix multiplication.

$$C_{i_1,j_1} = \sum_{k=1}^{d_2} A_{i_1,k} \, B_{k,j_1}$$

This operation extends naturally to higher-order tensors by summing over any shared index.

$$C_{i_1,\ldots,i_p,j_1,\ldots,j_q} = \sum_{k_1,\ldots,k_r} A_{i_1,\ldots,i_p,k_1,\ldots,k_r} \times B_{k_1,\ldots,k_r,j_1,\ldots,j_q}$$

where the indices $k_1, \ldots, k_r$ are summed over, representing the contracted dimensions shared by A and B. This operation generalizes matrix multiplication and inner product to higher-order tensors.

Graphical representation. Tensor contractions involving multiple tensors can be difficult to reason about. The graphical representation of tensors, as shown in Figure 2, provides a more intuitive way of visualizing them. In this representation, tensors are depicted as nodes and their indices as edges, with edges common to two tensors indicating a contraction.

Tensor decomposition. As the number of parameters grows exponentially with the tensor order, computing with them becomes costly. Tensor networks were originally introduced to efficiently represent high-order tensors by decomposing them into a network of lower-order tensors. The number of parameters of an order-n tensor $T \in \mathbb{R}^{d_0 \times d_1 \times d_2 \times \cdots \times d_n}$ is given by the product of its dimensions, $d_0 d_1 d_2 \cdots d_n$.

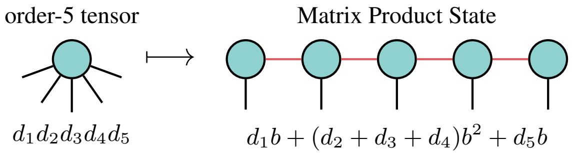


Figure 3: The decomposition of an order-5 tensor into a Matrix Product State (MPS). The red edges are called the bonds and their dimension is called the *bond dimension* b. Below the tensors, we show the formulas for the number of parameters required to represent the full order-5 tensor (bottom left) and its MPS decomposition (bottom right).

In many practical scenarios, representing a high-order tensor with all of its exponentially many parameters is unnecessary. Instead, the tensor can often be efficiently approximated or even exactly represented by decomposing it into a network of lower-order tensors. This decomposition, called a tensor network, greatly reduces the number of parameters and enables scalable computation.

A canonical example is the ground state of a quantum many-body system, which can be efficiently represented by a Matrix Product State (MPS) (Fannes et al., 1992), also known as a Tensor Train. An MPS expresses a high-order tensor as a sequence (or "train") of lower-order tensors connected by contracted indices, as illustrated in Figure 3. The contractions between neighboring tensors are called *bonds* and their dimensions are called *bond dimensions*. The dimension of each bond index is the *bond dimension* b, which controls the expressiveness and parameter count of the MPS. The total number of parameters in the MPS is

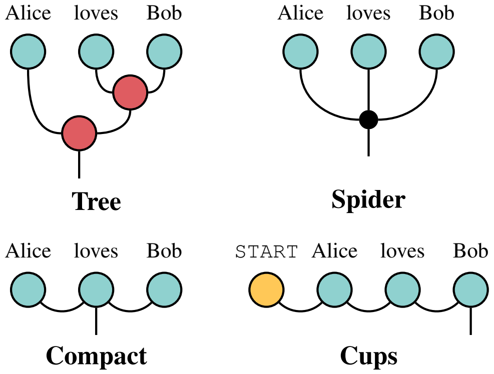


Figure 4: The four types of tensor networks considered in this paper: **Compact** and **Tree** are based on the CCG grammar, **Cups** preserves word order and **Spider** is a bag-of-words model. Each rectangle represents a node in the tensor network. The black dot in Spider is the copy node, which is operationally equivalent to element-wise multiplication.

$$(d_1 + d_n)b + \sum_{k=2}^{n-1} d_k b^2,$$

assuming all bond dimensions are equal to b. This is typically much smaller than the $d_1 d_2 \cdots d_n$ parameters required for a full tensor, making MPS an efficient representation for high-order tensors. For the rest of this paper, we will use a uniform dimension denoted by d and a uniform bond dimension denoted by b for simplicity. We denote the number of parameters in an MPS representation of an order-n tensor as

$$\#\text{MPS}(n, d, b) = \begin{cases} d, & n = 1 \\ 2db + (n-2)db^2, & n \geq 2 \end{cases} \tag{1}$$

Other than MPS, other common tensor network decompositions include the Tree Tensor Network (TTN) (Shi et al., 2006), which arranges tensors in a tree structure, and the Projected Entangled Pair State (PEPS) (Verstraete and Cirac, 2004), which arranges tensors in a 2D lattice. These decompositions are useful for different applications and can be adapted to specific data structures.

4 Methodology

Our main contribution is to replace CLIP's Transformer-based text encoder with a tensor network encoder, resulting in a new vision-language model we call **DisCoCLIP**. In DisCoCLIP, the text encoder constructs sentence embeddings using tensor networks that explicitly encode linguistic structure, while the image encoder remains the original CLIP vision transformer. By varying the layout of the tensor network, we can control the level of syntactic and semantic information captured in the text representation.

Given an image-caption pair, **DisCoCLIP** processes them in the following steps (see Figure 1):

1. The sentence is parsed to extract its syntactic structure.
2. A tensor network is constructed based on the parse tree, where each word is represented by a tensor node.
3. The tensor network is contracted to produce a fixed-size vector embedding for the entire sentence.
4. The image embedding is computed using a Vision Transformer (ViT).
5. The text and image embeddings are compared to compute a similarity score, which is used for training the model and for downstream evaluation.

For step 1, we use the state-of-the-art `BobcatParser` (Clark, 2021) from the `Lambeq` library (Kartsaklis et al., 2021) to obtain the Combinatory Categorial Grammar parse trees of the sentences (Ades and Steedman, 1982; Steedman, 1987, 2000) .

Combinatory Categorial Grammar (CCG) is a highly expressive formalism for modeling natural language syntax and semantics. In CCG, each word is assigned a syntactic category that reflects both its grammatical role and its combinatory potential. Categories are either atomic (such as noun phrase NP or sentence S) or functional, where functional categories specify how a word combines with its arguments. Functional types take the form Y/X or $Y\backslash X$, indicating that the word expects an argument of type X to its right (/) or left (\), and yields a result of type Y. For example, adjectives have type NP/NP, intransitive verbs have type $S\backslash NP$, and transitive verbs have type $(S\backslash NP)/NP$.

The combinatory rules of CCG allow for the composition of these categories to cancel out the functional types and yield a sentence S type. The two main rules are forward application (>) and backward application (<):

$$\frac{X/Y \quad Y}{X} > \qquad \frac{Y \quad X\backslash Y}{X} <$$

where X and Y are any CCG types. These rules allow for the composition of words into phrases and sentences, following the syntactic structure of the language. For example, the sequence "Alice loves Bob" can be reduced to a sentence S by first assigning the atomic category NP to both "Alice" and "Bob", and the functional category $(S\backslash NP)/NP$ to "love" and then applying the forward and backward application rules as follows:

$$\cfrac{\cfrac{Alice}{NP} \qquad \cfrac{\cfrac{loves}{(S\backslash NP)/NP} \quad \cfrac{Bob}{NP}}{S\backslash NP} >}{S} <$$

Other CCG rules include forward and backward composition, which are used to combine auxiliary verbs with their arguments, and forward and backward cross-composition, used to combine categories with long distance dependencies such as gapping. Another notable CCG rule is type-raising, which enables specific combinations of categories, e.g. from left to right. This feature helps the CCG align with Psycholinguistic theories. For instance, in English, it will allow categories to combine from left to right and form incremental parses that support theories of human sentence processing.

A distributional compositional (DisCo) semantics has been developed for CCG (Grefenstette and Sadrzadeh, 2011; Yeung and Kartsaklis, 2021; Wijnholds et al., 2020). This semantics assigns to a word w with a CCG category composed of n atomic categories a multilinear map f_w with n arguments

$$f_w \colon V_1 \times V_2 \times \cdots \times V_n \to V_{n+1}$$

Each V_i is a finite-dimensional vector space over the field of reals $\mathbb{R}$. Equivalently, f_w can be represented by a tensor of in the space

$$f_w \in V_1 \otimes V_2 \otimes \cdots \otimes V_{n+1}$$

Here, each atomic type corresponds to an index of the tensor. For example, a noun with the type NP is assigned a vector (order-1 tensor), while an adjective with the type NP/NP is assigned a linear map that takes a vector and returns a vector, which can be represented as a matrix (order-2 tensor). A transitive verb with the type $(S\backslash NP)/NP$ is assigned a bilinear map that takes two vectors and returns another vector, i.e. a cube (an order-3 tensor), and so on. For the general formulae of these representations, see (Maillard et al., 2014) and (Wijnholds et al., 2020).

Given the CCG parse tree, the word tensors are composed by performing tensor contractions that mirror the syntactic reductions specified by the tree. Each time a combinatory rule (such as forward or backward application) is applied in the parse, the corresponding word tensors are contracted along the appropriate indices. This process recursively combines the tensors according to the grammatical structure, ultimately yielding a single vector representation for the entire sentence. Such semantics was developed in (Maillard et al., 2014; Wijnholds et al., 2020) and leads to the **Compact** tensor network structure.

An alternative semantics presented in (Yeung and Kartsaklis, 2021) assigns to every word a vector and models the grammatical compositions (represented by CCG rules such as forward and backward application) by a shared order-3 tensor. This tensor acts as a universal composition operator of all compositional operators. This approach yields the **Tree** tensor network structure, where the parse tree topology is preserved but all internal nodes use the same composition tensor to combine their child representations.

4.1 Text Encoder Structures

We consider four types of tensor network structures: **Tree**, **Compact**, **Cups**, and **Spider**, as illustrated in Figure 4. Every tensor node in the networks is a trainable parameter, which is learned during the training process.

The **Tree** structure is based on the CCG parse tree of the sentence, where each word is represented as a vector node and an order-3 tensor is used to compose these word nodes to form non-terminal terms in the parse tree.

The **Compact** structure is a variant of the **Tree** structure, where every non-terminal node in the parse tree is absorbed by one of its parents, resulting in a more compact representation where some word nodes become higher-order tensors.

The **Cups** structure is a variant of Tensor Train (or MPS) where each word is an order-2 tensor, connected in a chain to preserve word order. The first word connects to a special `start` node while the last word outputs the sentence embedding.

The **Spider** structure implements a bag-of-words model, where each word is represented as a vector node and all word nodes are contracted through a special *copy node* to produce a single output vector. This copy node, of order n, is a tensor $C \in \mathbb{R}^{d^n}$ defined as

$$C_{i_1,i_2,\ldots,i_n} = \begin{cases} 1 & \text{if } i_1 = i_2 = \cdots = i_n, \\ 0 & \text{otherwise.} \end{cases}$$

Contracting $n-1$ indices of the copy node with $n-1$ word vectors yields their element-wise (Hadamard) product, producing a multiplicative bag-of-words sentence embedding.

Parameter count. Each tensor network structure has a different parameter count, determined by the number and order of word tensors and any composition tensors. Let $|V|$ be the vocabulary size. For **Compact**, let $|V^r|$ be the number of words with order-r tensors, and $\#\text{MPS}(r, d, b)$ the parameter count for an order-r MPS (see Eq. (1)). Table 1 summarizes the counts.

Structure	Words	Composition
Tree	$\lvert V \rvert d$	$2db + db^2$
Compact	$\sum_r \lvert V^r \rvert \#\text{MPS}(r, d, b)$	0
Spider	$\lvert V \rvert d$	0
Cups	$\lvert V \rvert d^2$	d

Table 1: Number of parameters for each tensor network structure.

5 Contrastive Learning

We train **DisCoCLIP** using contrastive learning, where the image encoder f (frozen CLIP) and the tensor network text encoder g map image-caption pairs (x, y) to embeddings $(\mathbf{x}, \mathbf{y})$. The goal is to bring true (positive) pairs closer and push (negative) mismatched pairs apart in the joint embedding space. For a batch of B positive pairs, all non-matching image-caption combinations in the batch ($B(B-1)$) serve as negatives, following the in-batch negative sampling of CLIP (Radford et al., 2021). These are considered *easy* negatives, as opposed to more challenging, hand-crafted *hard* negatives.

We use the widely adopted InfoNCE loss (van den Oord et al., 2018) to train the model. Given a batch of B image-caption pairs with embeddings $\mathbf{x}_i = f(x_i)$ and $\mathbf{y}_i = g(y_i)$, the InfoNCE loss is

$$\mathcal{L} = -\sum_{i=1}^{B} \log \frac{\exp(s(\mathbf{x}_i, \mathbf{y}_i)/\tau)}{\sum_{j=1}^{B} \exp(s(\mathbf{x}_i, \mathbf{y}_j)/\tau)}, \quad (2)$$

where τ is a temperature parameter and $s(\mathbf{x}, \mathbf{y})$ is the cosine similarity between the image embedding

$\mathbf{x}$ and the caption embedding $\mathbf{y}$. Here, the numerator measures similarity for positive (matching) pairs ($i = j$), while the denominator includes all pairs in the batch, serving as negatives when $i \neq j$. The loss thus encourages higher similarity for true pairs and lower for mismatched ones.

We evaluate our approach on two key benchmarks for vision-language understanding: SVO-Probes and ARO. The SVO-Probes dataset is designed to test whether models can distinguish fine-grained changes in the image which corresponds to variations in subject, verb, or object. The task is to determine which of the two images correctly matches a given caption. In contrast, the ARO (Attribution, Relation, and Order) dataset assesses a model's ability to correctly compose meanings in a sentence. The task is to determine which of the two captions correctly describes a given image. Together, these datasets provide a comprehensive evaluation of both compositional and structural language understanding in multimodal models.

5.1 SVO-Probes

For SVO-Probes, we prompted the language model `Llama-3.2-3B` (Grattafiori et al., 2024) to correct grammatical and spelling mistakes as the original dataset contained errors from crowdsourced captions. The exact prompt used can be found in the Appendix A. The images in the SVO-Probes dataset were not available for download from the official repository; therefore, we attempted to download them from the Internet using the provided URLs on 2 May 2025. However, many of the URLs were no longer active, and we were only able to download 8,984 images of the total 14,097 images in the dataset, resulting in a reduction of the dataset size from 36,841 to 20,458 entries.

To ensure a reasonable train-test vocabulary overlap, we filtered out entries that contained words that appeared fewer than 50 times in the entire dataset, yielding 8,984 image-caption pairs split 60/20/20 for training, validation, and test, with no image overlap between splits. We also introduced a new dataset: SVO-Swap. This is a set of 95 evaluation pairs created by swapping subjects and objects (when both refer to humans or animals) in SVO-Probes captions.

The SVO-Probes benchmark is divided into three subsets: Subject, Verb, and Object. Each of these corresponds to the specific component of the sentence that differs between the two alternatives. This structure enables a fine-grained evaluation of the model's ability to distinguish changes in the linguistic roles of the words within a caption.

5.2 ARO

The ARO dataset consists of four different subsets: Visual Genome Attribution (VG-A), Visual Genome Relation (VG-R), COCO Order and Flickr Order. The way these subsets are constructed was to first gather a set of positive image-caption pairs, and then apply a certain modification to the captions to form negative captions. In the VG-A subset, positive pairs are chosen to be images with two objects and each gets an attribute. For example *the silver fork and the round plate* contains a fork that is silver, and a plate that is round. The corresponding negative caption would be *the round fork and the silver plate*, where the attributes for the two objects are swapped. For the VG-R subset, positive pairs are images with a relation involving two objects. For example For ARO-Attribution (with 28,748 entries) and ARO-Relation (with 23,937 entries), we used a 70/15/15 split without frequency filtering, as vocabulary overlap was sufficient.

5.3 Training

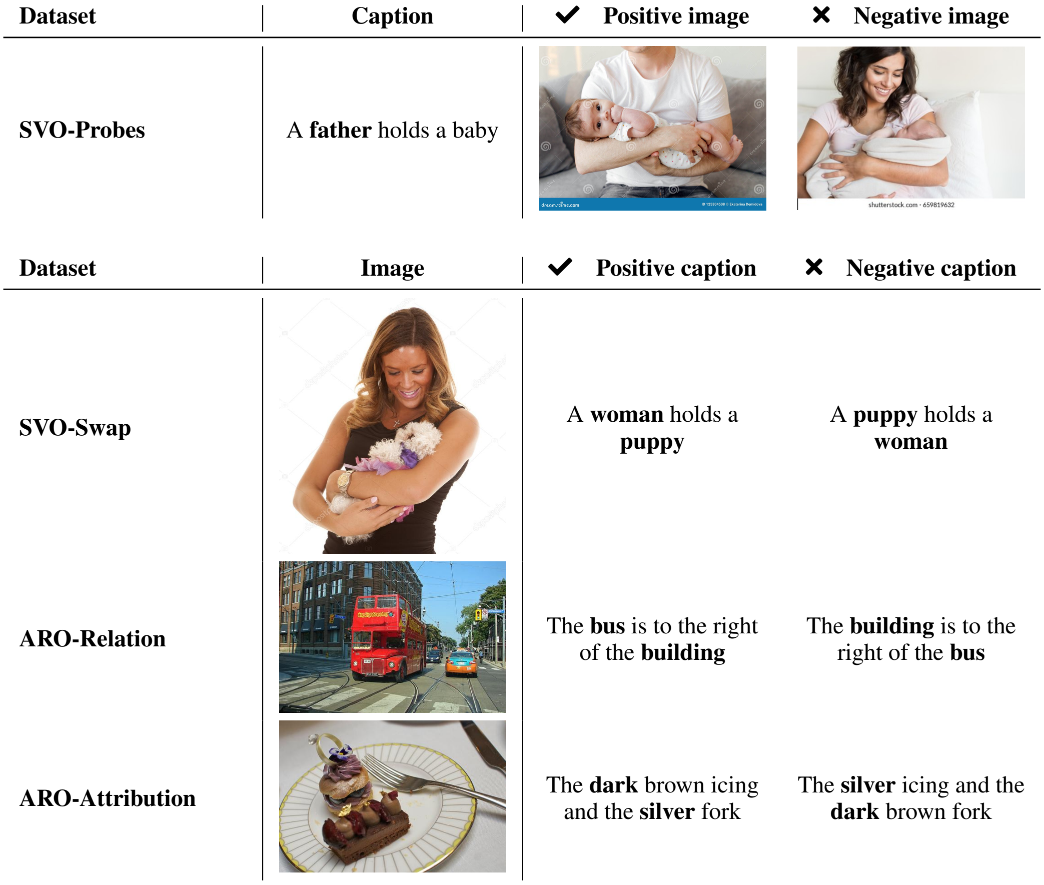
For each structure (**Tree**, **Compact**, **Spider** and **Cups**), we trained the tensors for 10 epochs, using the `AdamW` optimizer (Loshchilov and Hutter, 2019) with a learning rate of 10^{-3}, a weight decay of 10^{-2} and a batch size of 64. We also experimented with bond dimensions 2, 5, 10, 15 and 20 in the MPS decomposition. Training was performed on an Apple M1 MacBook with 16GB RAM, utilizing the PyTorch Metal Performance Shaders (`mps`) backend to accelerate tensor operations on the GPU. Each epoch required several minutes, and the total training time for all experiments was approximately one day. The code used for the experiments is available at github.com/kinianlo/discoclip.

5.4 Results

Table 2 reports our performance on SVO-Probes and ARO. Although BLIP achieves the highest raw scores on SVO-Probes subsets(Subjects (91.88), Verbs (88.58), Objects (96.37)), our **Compact** model remains a strong second overall (83.55) and is the clear leader among non-BLIP approaches. Notably, **Compact** scores higher on Verbs (82.42) than on Subjects (80.74), reversing the typical trend seen in all other models and underscoring its structure-aware design for modeling action semantics. On the SVO-Swap benchmark, **Compact**

	SVO-Probes				SVO-Swap	ARO	
	Subject	Verb	Object	Overall		Attribution	Relation
Spider	83.29	76.48	86.64	80.95	50.00	50.00	50.00
Cups	74.25	75.36	86.83	78.35	84.21	63.07	52.68
Tree	89.79	79.40	85.88	83.66	47.37	55.11	52.36
Compact	80.74	82.42	87.79	83.55	**93.68**	70.01	**55.81**
CLIP (ViT-B-32)	82.83	77.60	90.08	82.36	57.89	61.00	51.53
OpenCLIP (ViT-B-32)	85.15	81.41	93.51	85.71	63.16	59.13	50.71
BLIP (itm-base-coco)	**91.88**	**88.58**	**96.37**	**91.56**	36.84	**78.46**	52.90

Table 2: Results on the SVO-Probes, SVO-Swap and the ARO datasets. Best accuracies are bolded for each subset.



Dataset	Caption	✓ Positive image	✗ Negative image
SVO-Probes	A **father** holds a baby		

Dataset	Image	✓ Positive caption	✗ Negative caption
SVO-Swap		A **woman** holds a **puppy**	A **puppy** holds a **woman**
ARO-Relation		The **bus** is to the right of the **building**	The **building** is to the right of the **bus**
ARO-Attribution		The **dark** brown icing and the **silver** fork	The **silver** icing and the **dark** brown fork

Figure 5: Example entries from the datasets used in this work.

excels with 93.68, highlighting its robustness to argument perturbations. Finally, on ARO, **Compact** outperforms every model on Relation attribution (55.81) and closely matches BLIP on Attribution (70.01), demonstrating that embedding syntactic structure as an inductive bias without hard-negative training yields consistently strong relational reasoning and verb understanding.

It is noteworthy that although **BLIP** achieved the highest overall accuracy on SVO-Probes (91.56), it performed poorly on our new SVO-Swap benchmark (36.84). The underlying causes of this discrepancy remain under investigation.

By contrast, the baseline models **Spider** and **Cups** deliver the poorest performance, underscoring that correct structural encoding is essential for compositional understanding. As a bag-of-words model, **Spider** produces identical representations for both candidate captions in SVO-Swap and ARO, resulting in a flat 50 percent accuracy on these tasks. This failure further illustrates the necessity of incorporating explicit linguistic structure rather than relying solely on word co-occurrence.

Parameter Efficiency As shown in Table 1, our **Compact** text encoder requires only 537,600 parameters on the SVO-Probes benchmark—over two

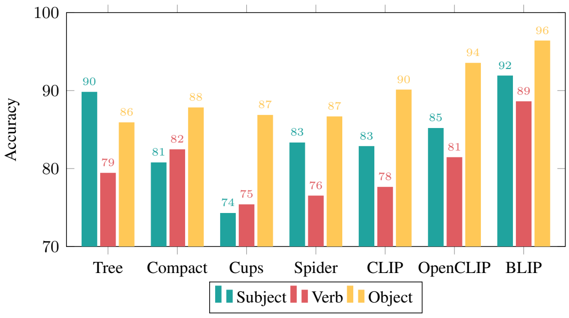


Figure 6: Performance of models on the SVO-Probes Subject, Verb, and Object subsets.

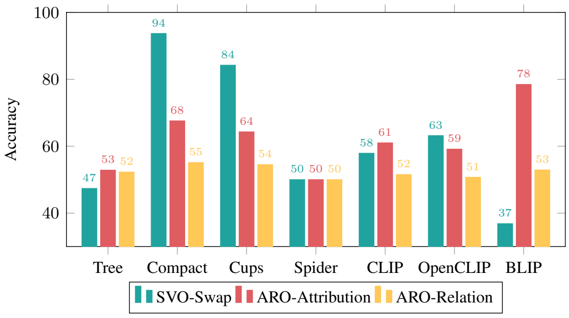


Figure 7: Performance of models on the SVO-Swap and ARO Attribution and Relation benchmarks.

orders of magnitude fewer than CLIP's 63,428,097 and BLIP's 137 258 496—benefiting from its tensor-train factorization and the relatively small vocabulary size. On the ARO benchmark, **Compact** uses 28,309,504 parameters—approximately two times fewer than CLIP's text encoder—while still outperforming CLIP in both attribution and relation accuracy.

Model	**SVO**	**ARO**
Spider	$55,296$	$735,744$
Cups	$1,659,392$	$14,715,392$
Tree	$185,856$	$797,184$
Compact	$537,600$	$28,309,504$
CLIP	63,428,097	
OpenCLIP	63,428,097	
BLIP	137,258,496	

Table 3: Parameter counts for each text encoder model.

6 Conclusion

In this work, we introduced **DisCoCLIP**, a vision-language model that replaces the standard Transformer-based text encoder with a structure-informed tensor network. By leveraging the compositional layouts of tensor networks inspired by compositional distributional semantics and quantum-inspired tensor decompositions, our approach explicitly encodes linguistic structure and achieves competitive performance on challenging multimodal benchmarks such as SVO-Probes and ARO. Our experiments demonstrate that structure-aware tensor networks, particularly the **Compact** model that was a dense variant of the syntactic parse tree, can match or surpass classical neural models in tasks requiring fine-grained understanding of sentence structure. Our model also uses significantly fewer number of parameters in comparison to Transformer-based models such as CLIP. These results highlight the potential of tensor network architectures as interpretable and parameter-efficient alternatives for multimodal learning. Future work will explore scaling these models to larger datasets, working with complex datasets such as Winoground (Thrush et al., 2022), exploring the quantum connections and training circuit ansatze, and extending the approach to more complex linguistic phenomena.

7 Limitations

A limitation of this work is its evaluation on smaller, curated datasets rather than the web-scale data used to train many contemporary vision-language models. The SVO-Swap benchmark comprises only 95 evaluation pairs. Consequently, the performance reported on this task is not statistically robust.

Our pipeline introduces a dependency on the CCG parser. Errors from the parser can propagate through the pipeline, resulting in ill-formed tensor networks and inaccurate semantic representations. In future work, longer and more complicated sentences can be tested to see how parsing errors affect the performance of DisCoCLIP.

The image encoder was kept frozen during training, meaning the text encoder learned to align with a fixed set of visual features rather than co-adapting with the image encoder. While this design choice effectively isolates the contribution of the text encoder, training both the text and image encoder could potentially yield further performance improvements.

References

Anthony E. Ades and Mark J. Steedman. 1982. On the order of words. *Linguistics and Philosophy*, 4:517–558.

Seyed S. Ahromi and Román Orús. 2024. Variational tensor neural networks for deep learning. *Scientific Reports*, 14:19017.

Borja Aizpurua, Samuel Palmer, and Román Orús. 2025. Tensor networks for explainable machine learning in cybersecurity. *Neurocomputing*, 639:130211.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193.

Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature*, 549(7671):195–202.

Guillaume Bouchard, Jason Naradowsky, Sebastian Riedel, Tim Rocktäschel, and Andreas Vlachos. 2015. Matrix and tensor factorization methods for natural language processing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing: Tutorial Abstracts*, pages 16–18.

Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, and Danilo P. Mandic. 2016. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. *Foundations and Trends in Machine Learning*, 9(4-5):249–429.

Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, and Danilo P. Mandic. 2017.

Stephen Clark. 2021. Something old, something new: Grammar-based ccg parsing with transformer models. *CoRR*, abs/2109.10044.

M. Fannes, B. Nachtergaele, and R. F. Werner. 1992. Finitely correlated states on quantum spin chains. *Communications in Mathematical Physics*, 144(3):443–490.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404.

C. Harvey, R. Yeung, and K. Meichanetzidis. 2025. Sequence processing with quantum-inspired tensor networks. *Scientific Reports*, 15:7155.

Lisa Anne Hendricks and Aida Nematzadeh. 2021. Probing image–language transformers for verb understanding. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3635–3644, Online. Association for Computational Linguistics.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. 2021. Openclip.

Kenan Jiang, Xuehai He, Ruize Xu, and Xin Wang. 2024. ComCLIP: Training-free compositional image and text matching. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6639–6659, Mexico City, Mexico. Association for Computational Linguistics.

Dimitri Kartsaklis, Ian Fan, Richie Yeung, Anna Pearson, Robin Lorenz, Alexis Toumi, Giovanni de Felice, Konstantinos Meichanetzidis, Stephen Clark, and Bob Coecke. 2021. lambeq: An Efficient High-Level Python Library for Quantum NLP. *arXiv preprint arXiv:2110.04236*.

Martha Lewis, Nihal Nayak, Peilin Yu, Jack Merullo, Qinan Yu, Stephen Bach, and Ellie Pavlick. 2024. Does CLIP bind concepts? probing compositionality in large image models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1487–1500, St. Julian's, Malta. Association for Computational Linguistics.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*.

Wei Li, Zhen Huang, Xinmei Tian, Le Lu, Houqiang Li, Xu Shen, and Jieping Ye. 2024. Interpretable composition attribution enhancement for visio-linguistic compositional understanding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14616–14632, Miami, Florida, USA. Association for Computational Linguistics.

Ning Liu, Benyu Zhang, Jun Yan, Zheng Chen, Wenyin Liu, Fengshan Bai, and Leefeng Chien. 2005. Text representation: From vector to tensor. In *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM)*, pages 725–728.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*. Poster. Also available as arXiv:1711.05101.

Jean Maillard, Stephen Clark, and Edward Grefenstette. 2014. A type-driven tensor-based semantics for CCG. In *Proceedings of the EACL 2014 Workshop on Type*

Theory and Natural Language Semantics (TTNLS), pages 46–54.

Jacob E Miller, Guillaume Rabusseau, and John Terilla. 2021. Tensor networks for probabilistic sequence modeling. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3430–3440. PMLR.

Saba Nazir and Mehrnoosh Sadrzadeh. 2024. How does an adjective sound like? exploring audio phrase composition with textual embeddings. In *Proceedings of the 2024 CLASP Conference on Multimodality and Interaction in Language Learning*, pages 13–18, Gothenburg, Sweden. Association for Computational Linguistics.

Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry Vetrov. 2015. Tensorizing neural networks. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 442–450, Cambridge, MA, USA. MIT Press.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*.

Aditya S. Rao, Erik Bekkers, and 1 others. 2020. Tensor networks for medical image classification. *Proceedings of Machine Learning Research (PMLR)*, 121:123–134.

Chase Roberts, Charles Casert, and 1 others. 2019. Tensornetwork for machine learning. *Quantum Science and Technology*, 4(3):035002.

Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. 2015. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185.

Luciano Serafini and Artur d'Avila Garcez. 2017. Logic tensor networks for semantic image interpretation. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 26:1875–1881.

Yaoyun Shi, Luming Duan, and Guifré Vidal. 2006. Classical simulation of quantum many-body systems with a tree tensor network. *Physical Review A*, 74(2):022320.

Mark Steedman. 1987. Combinatory grammars and parasitic gaps. *Natural Language & Linguistic Theory*, 5(3):403–439.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

E Miles Stoudenmire. 2018. Learning relevant features of data with multi-scale tensor networks. *Quantum Science and Technology*, 3(3):034003.

E. Miles Stoudenmire and David J. Schwab. 2016. Supervised learning with tensor networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 29.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. In *CVPR*, pages 5228–5238. IEEE.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Frank Verstraete and Juan I. Cirac. 2004. Renormalization algorithms for quantum-many body systems in two and higher dimensions. *arXiv Preprint*.

Hadi Wazni, Kin Ian Lo, and Mehrnoosh Sadrzadeh. 2024. VerbCLIP: Improving verb understanding in vision-language models with compositional structures. In *Proceedings of the 3rd Workshop on Advances in Language and Vision Research (ALVR)*, pages 195–201.

Steven R. White. 1992. Density matrix formulation for quantum renormalization groups. *Physical Review Letters*, 69(19):2863–2866.

Gijs Wijnholds, Mehrnoosh Sadrzadeh, and Stephen Clark. 2020. Representation learning for type-driven composition. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 313–324.

Richie Yeung and Dimitri Kartsaklis. 2021. A CCG-based version of the DisCoCat framework. In *Proceedings of the 2021 Workshop on Semantic Spaces at the Intersection of NLP, Physics, and Cognitive Science (SemSpace)*, pages 20–31, Groningen, The Netherlands. Association for Computational Linguistics.

Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. 2023. When and why vision-language models behave like bags-of-words, and what to do about it? In *International Conference on Learning Representations*.

Lipeng Zhang, Peng Zhang, Xindian Ma, Shuqin Gu, Zhan Su, and Dawei Song. 2019. A generalized language model in tensor space. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7450–7458. AAAI Press.

A Prompt for Grammatical Correction for SVO-Probes

Listing 1: Prompt provided to Llama-3.2-3B-Instruct for grammatical correction for caption in the SVO-Probes dataset.

```
### System
You are a grammar assistant expert in
   Combinatory Categorial Grammar.
```

```
### Variables
Subject: {subj}
Verb: {verb}
Object: {obj}

### Task
Turn the user’s caption fragment into a
    single English sentence that:
- Is grammatically correct
- Has a valid CCG parse that leads to a
    sentence output
- Has no spelling errors
- Has a main verb {verb} in simple
    present tense only
- Has the subject ({subj}) first,
    followed by the verb ({verb}), then
    the object ({obj})

Additional rules:
- If the main verb is not {verb}, you
    may remove parts of the user’s input
- If the user’s input is a question,
    convert it into an affirmative
    sentence.

If it’s already correct, repeat it
    verbatim.
Respond **only** with the final sentence
    .

### Example
Input: Girl standing in the grass.
Output: The girl stands in the grass.

Input: A person is telling the boy to
    sit on the chair.
Output: The boy sits on the chair.

Input: The player backhands when he
    plays tennis.
Output: The player plays a backhand when
     he plays tennis.

Input: Can we take the kid for a walk on
     the beach?
Output: The kid walks on the beach.

Input: Is this person resting under the
    tree?
Output: The person rests under the tree.

### User
{input_sentence}
```