Vector-based Models of Semantic Composition

Jeff Mitchell and Mirella Lapata, 2008

Composition in Distributional Models of Semantics

Jeff Mitchell and Mirella Lapata, 2010

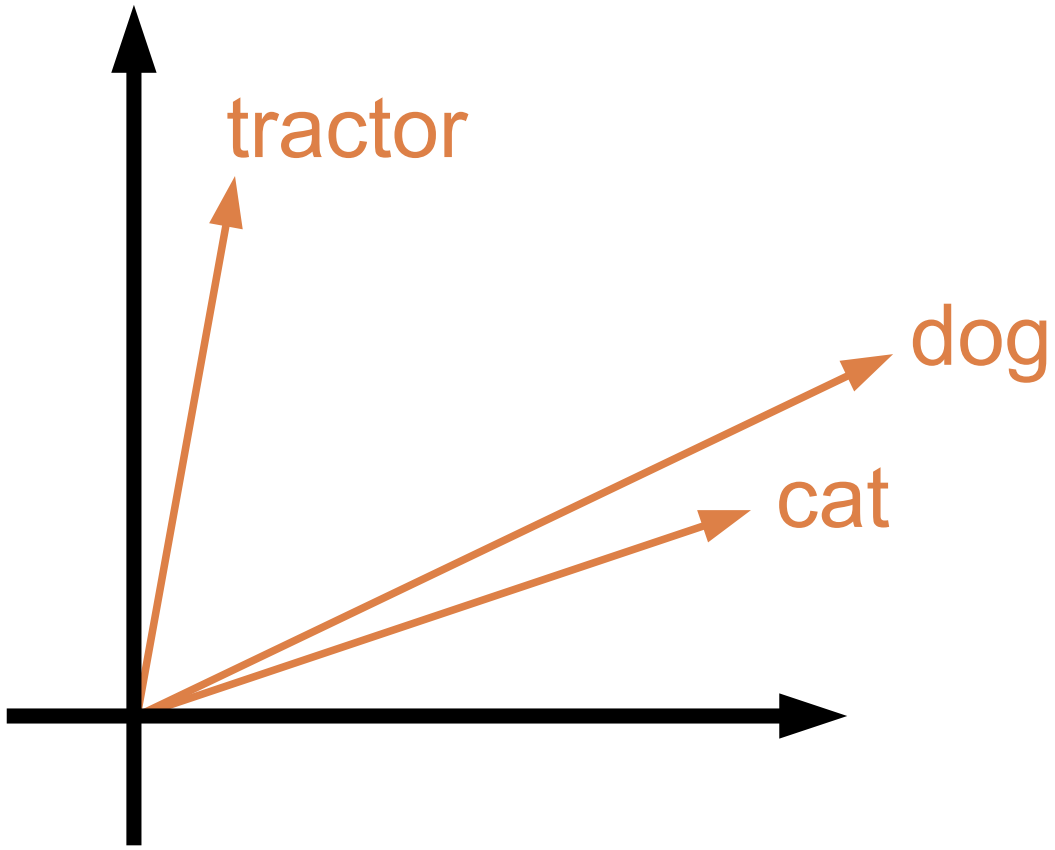
Distributional Hypothesis

"Words occurring within similar contexts are semantically similar"

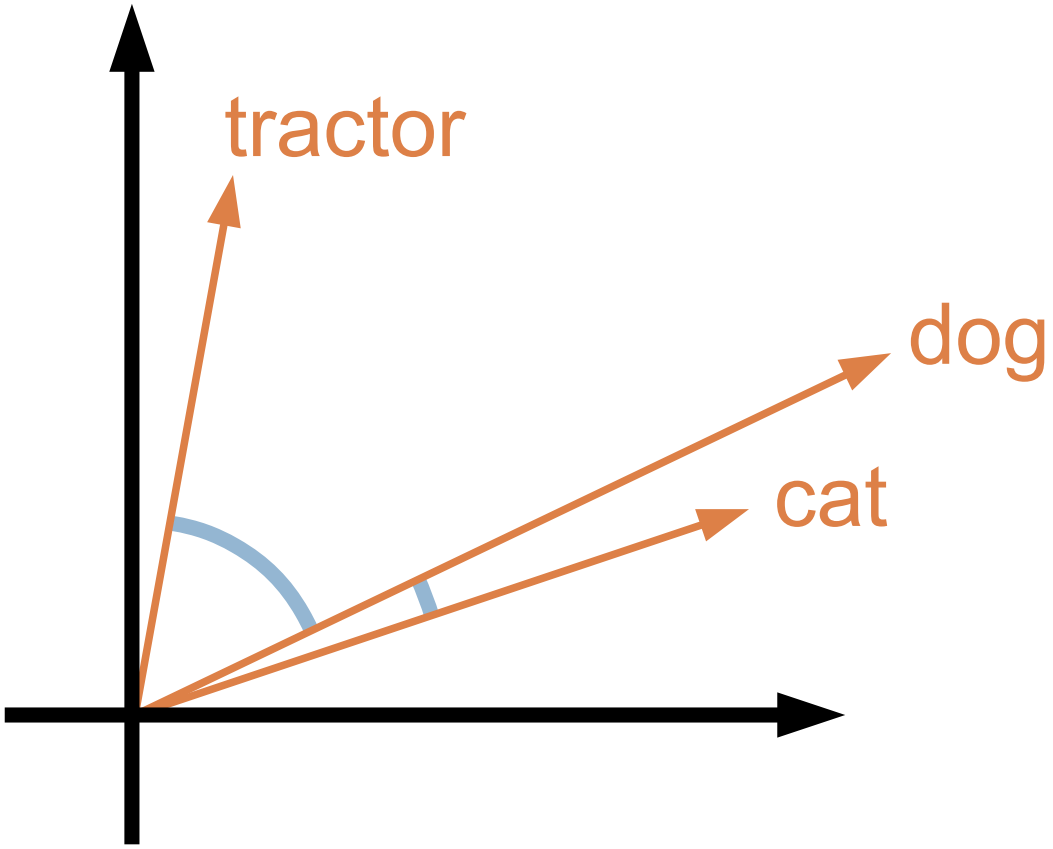
What is a context?

- Window of words
  - Co-occurrence frequencies
- Document
  - LDA topic assignments

Semantic Vector Spaces



Semantic Vector Spaces



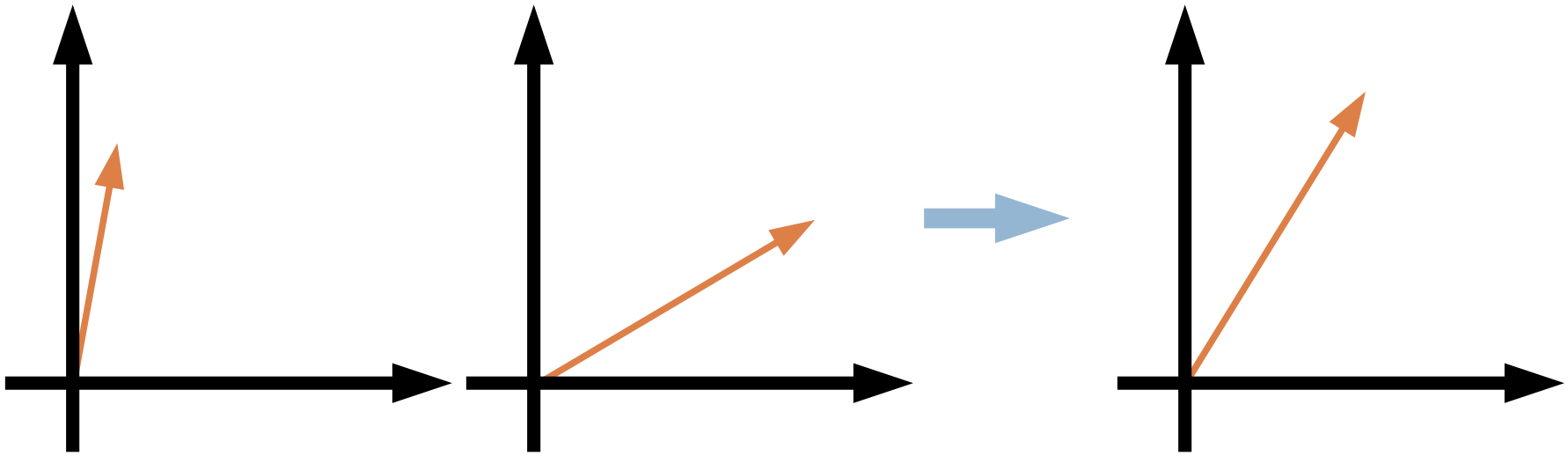
Aim of the paper

How can we combine distributional vectors?

General Framework

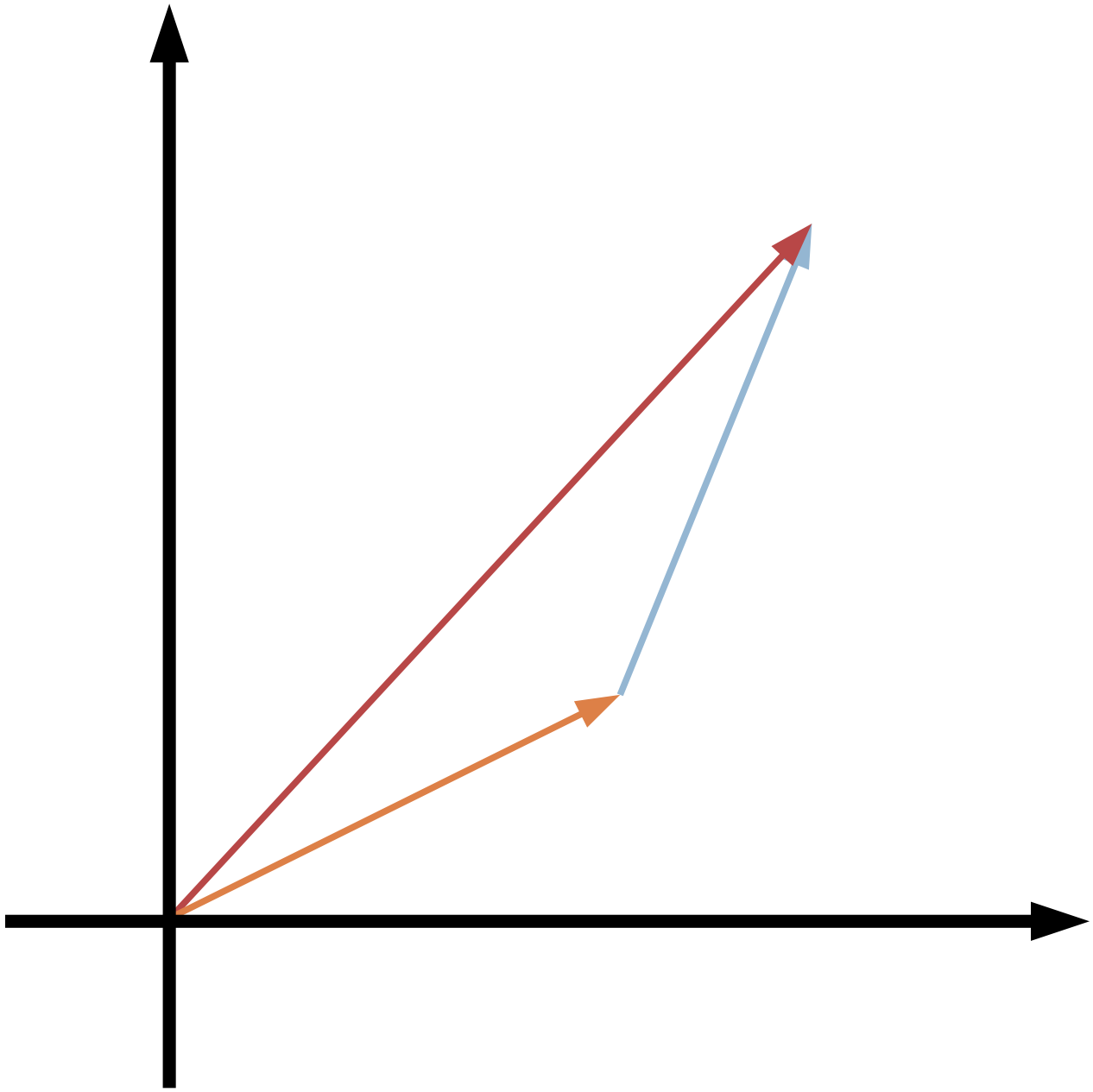
$$f : V \times V \rightarrow V$$

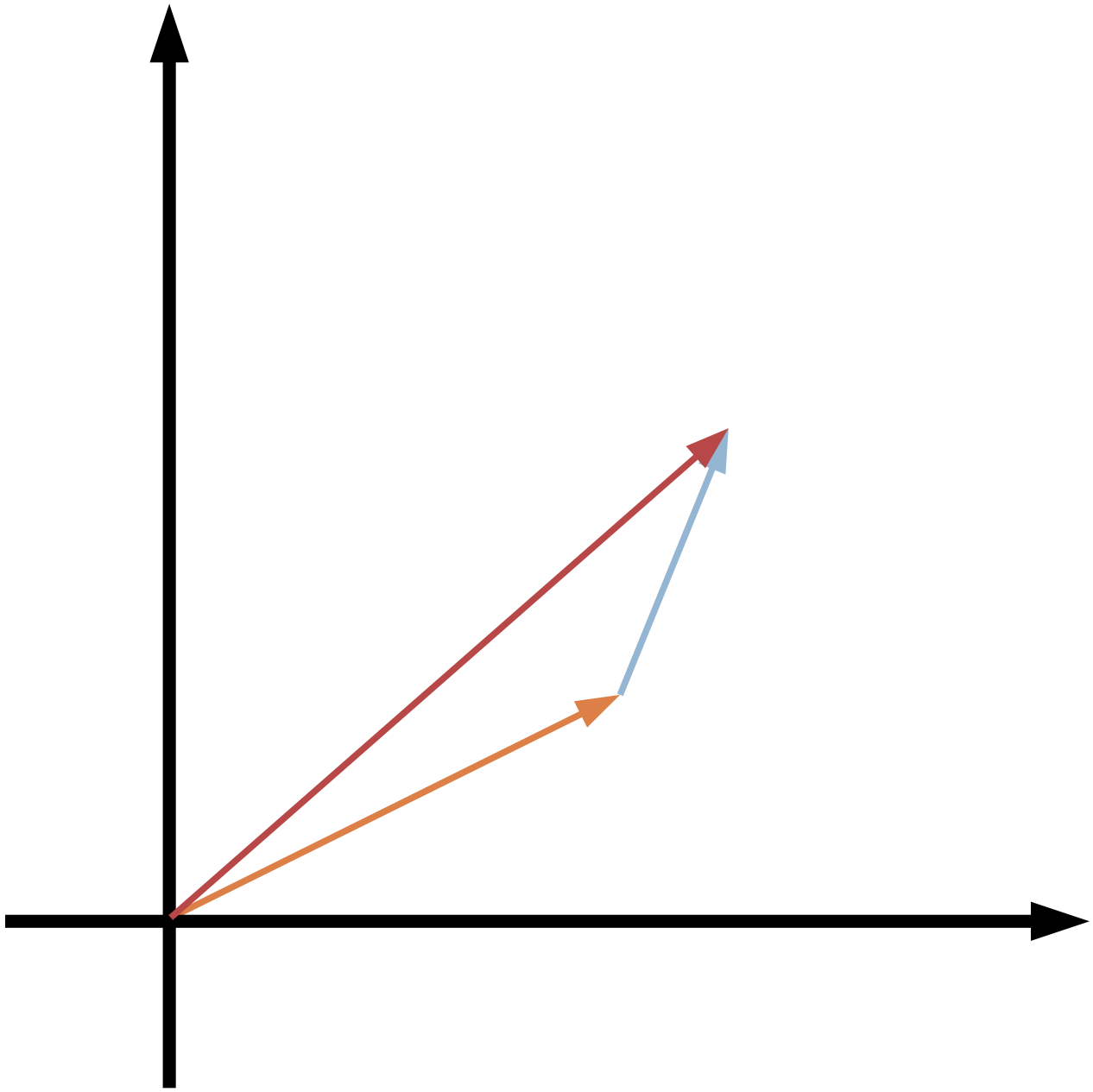
General Framework

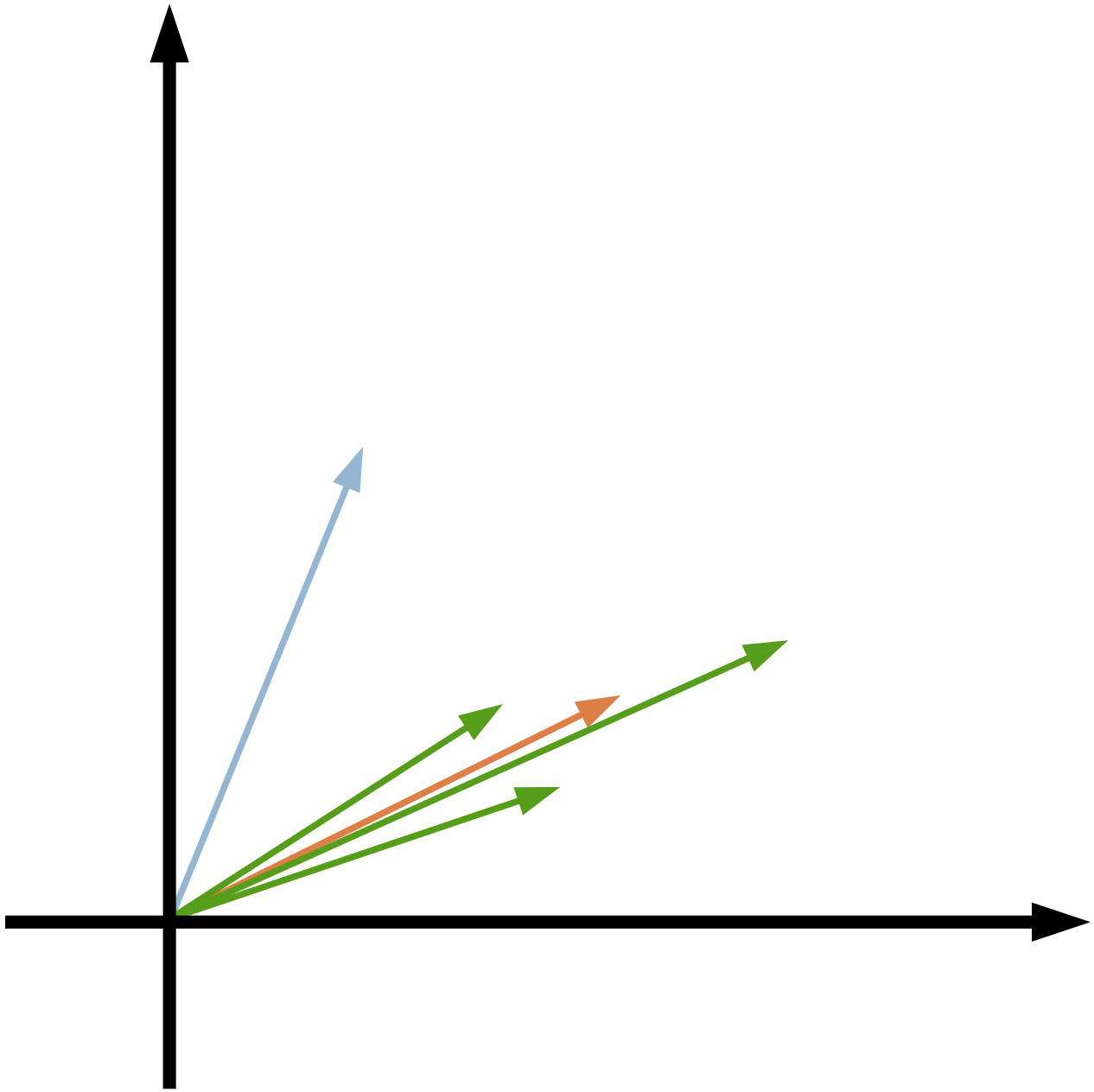


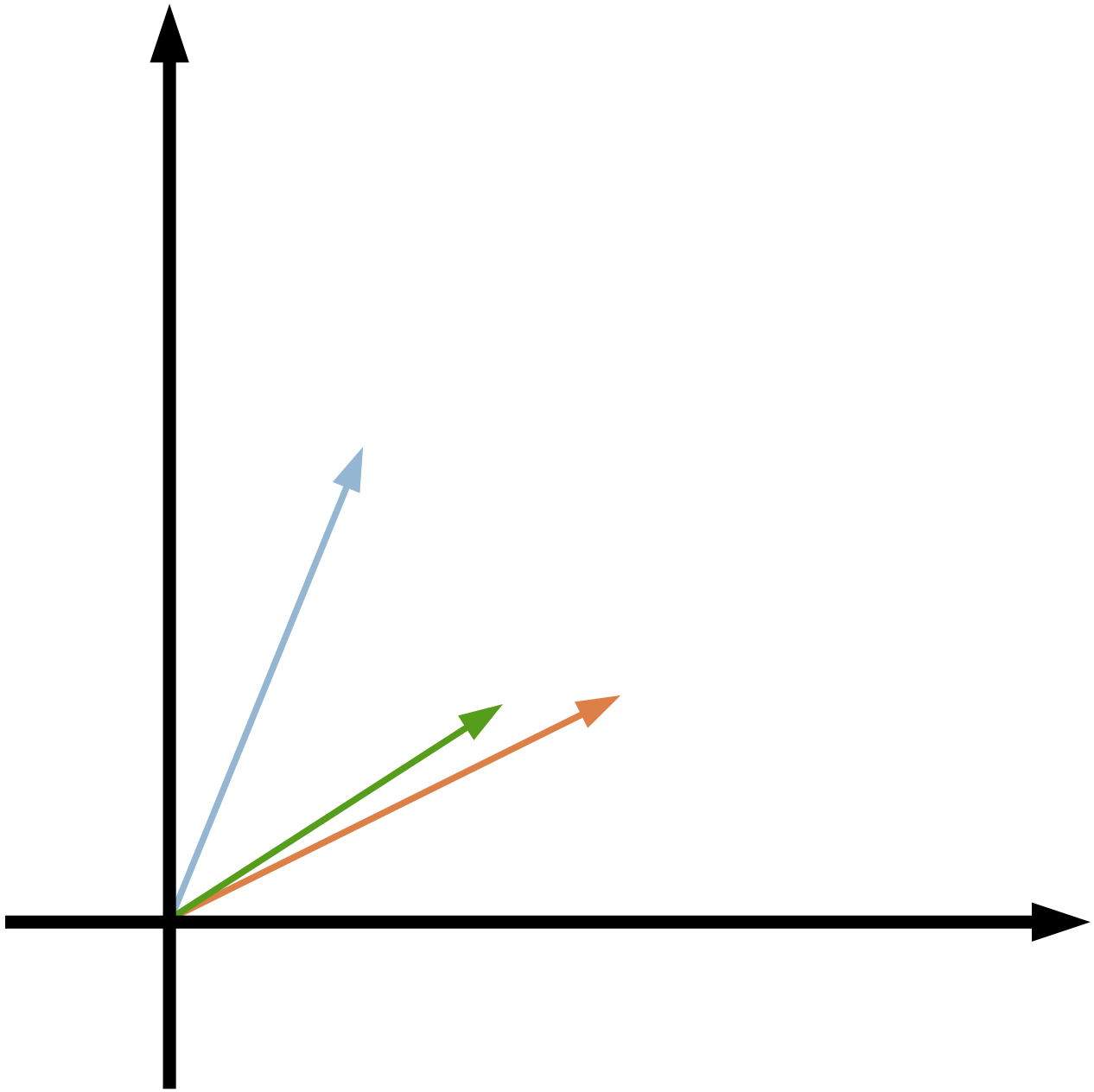
Composition Functions

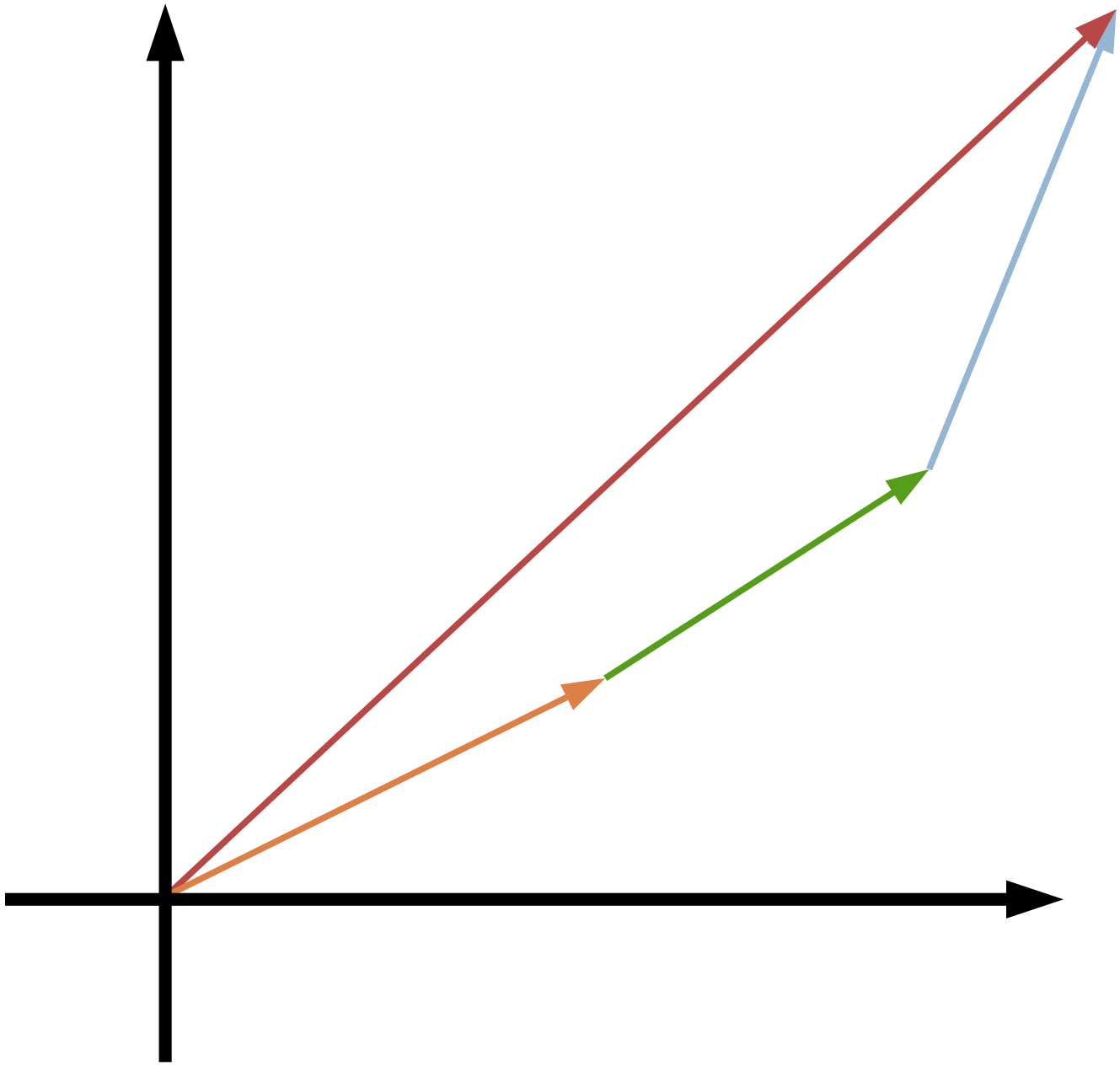
Additive	$p_i = u_i + v_i$
Weighted Additive	$p_i = \alpha u_i + \beta v_i$
Kintsch	$p_i = u_i + v_i + n_i$
Dilation	$p_i = (\lambda - 1)(u.v)u_i + \lvert u \rvert^2 v_i$
Multiplicative	$p_i = u_i v_i$
Tensor Product	$p_{ij} = u_i v_j$
Circular Convolution	$p_i = \Sigma_j u_j v_{i-j}$

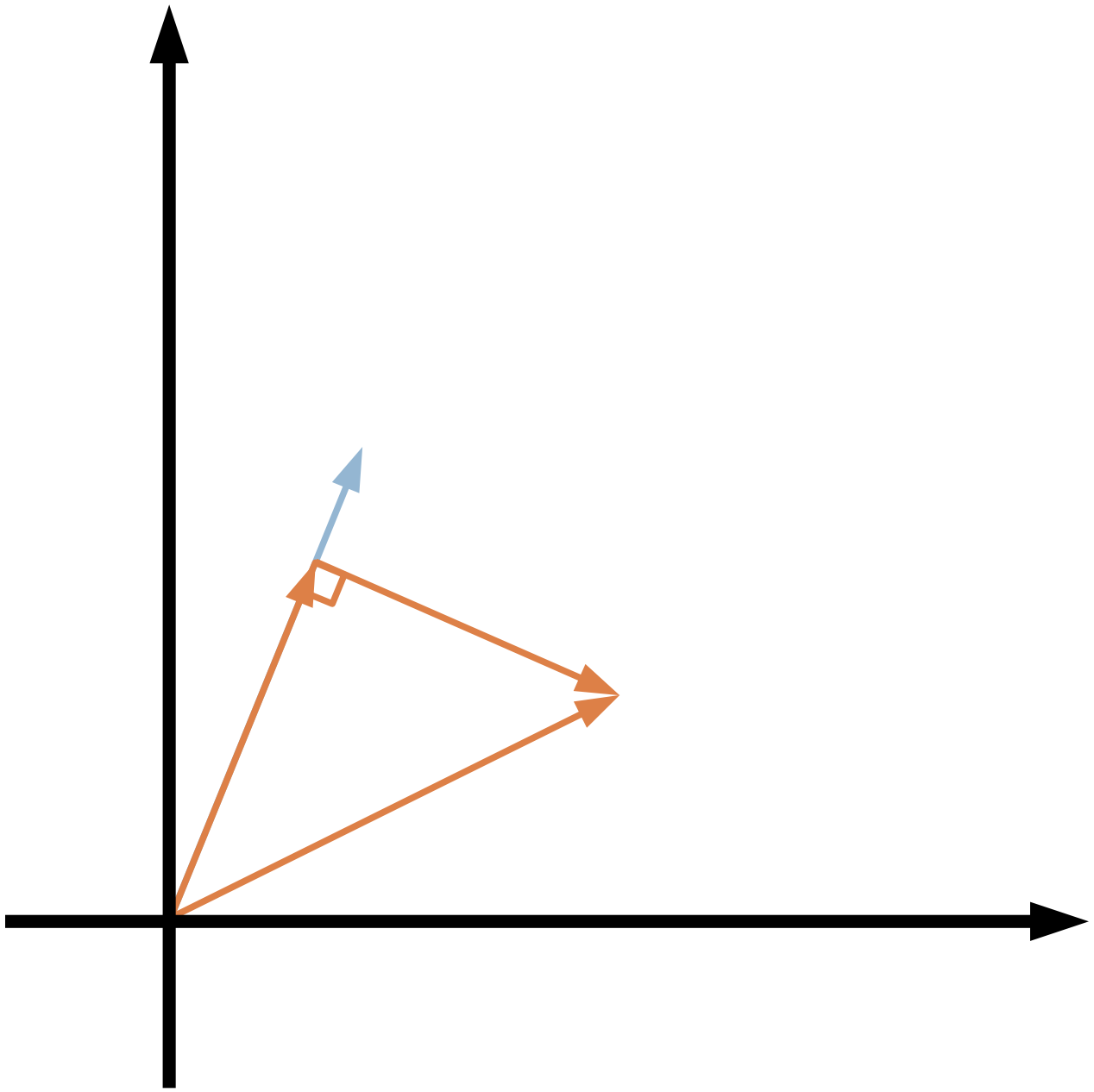


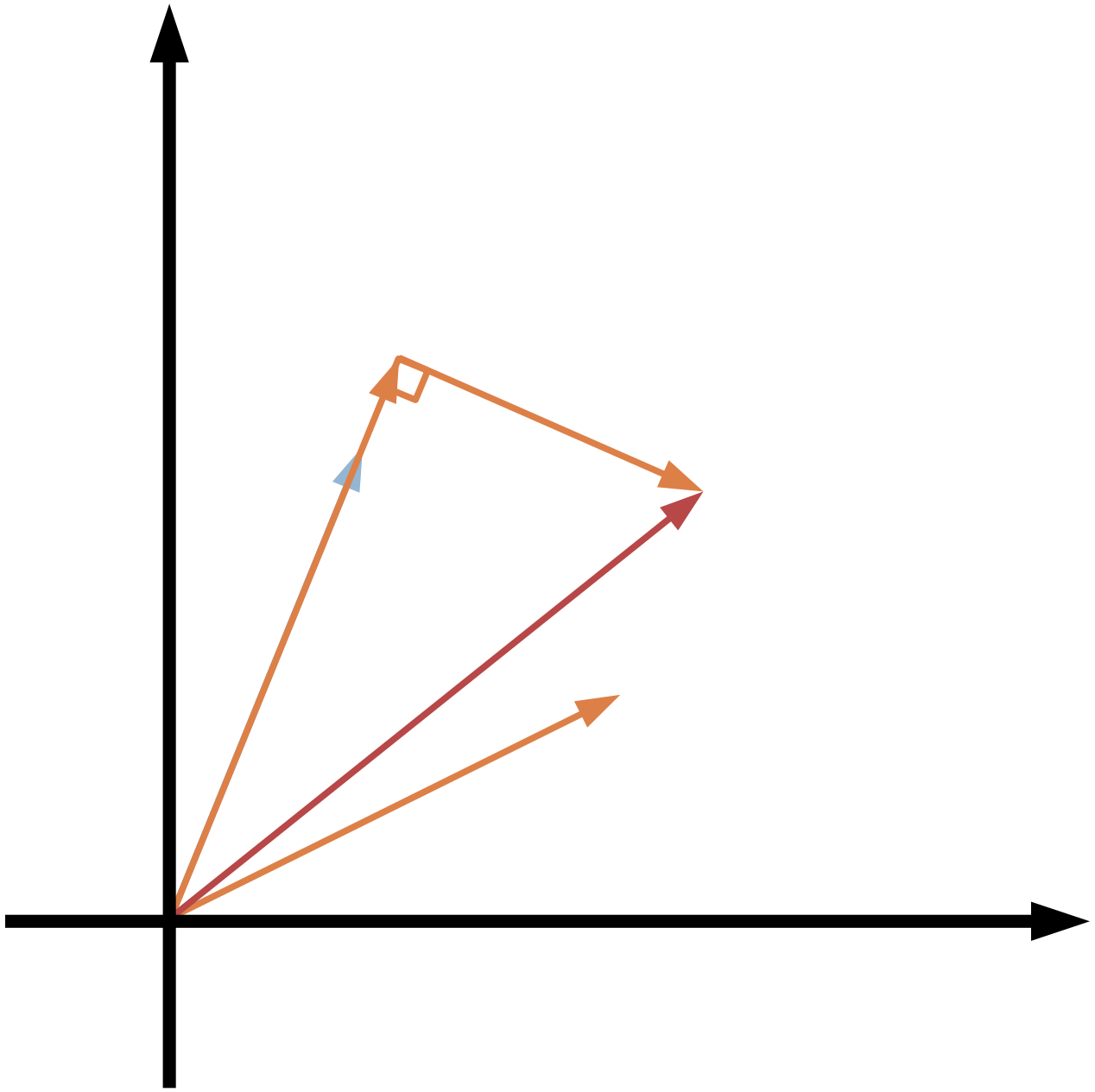


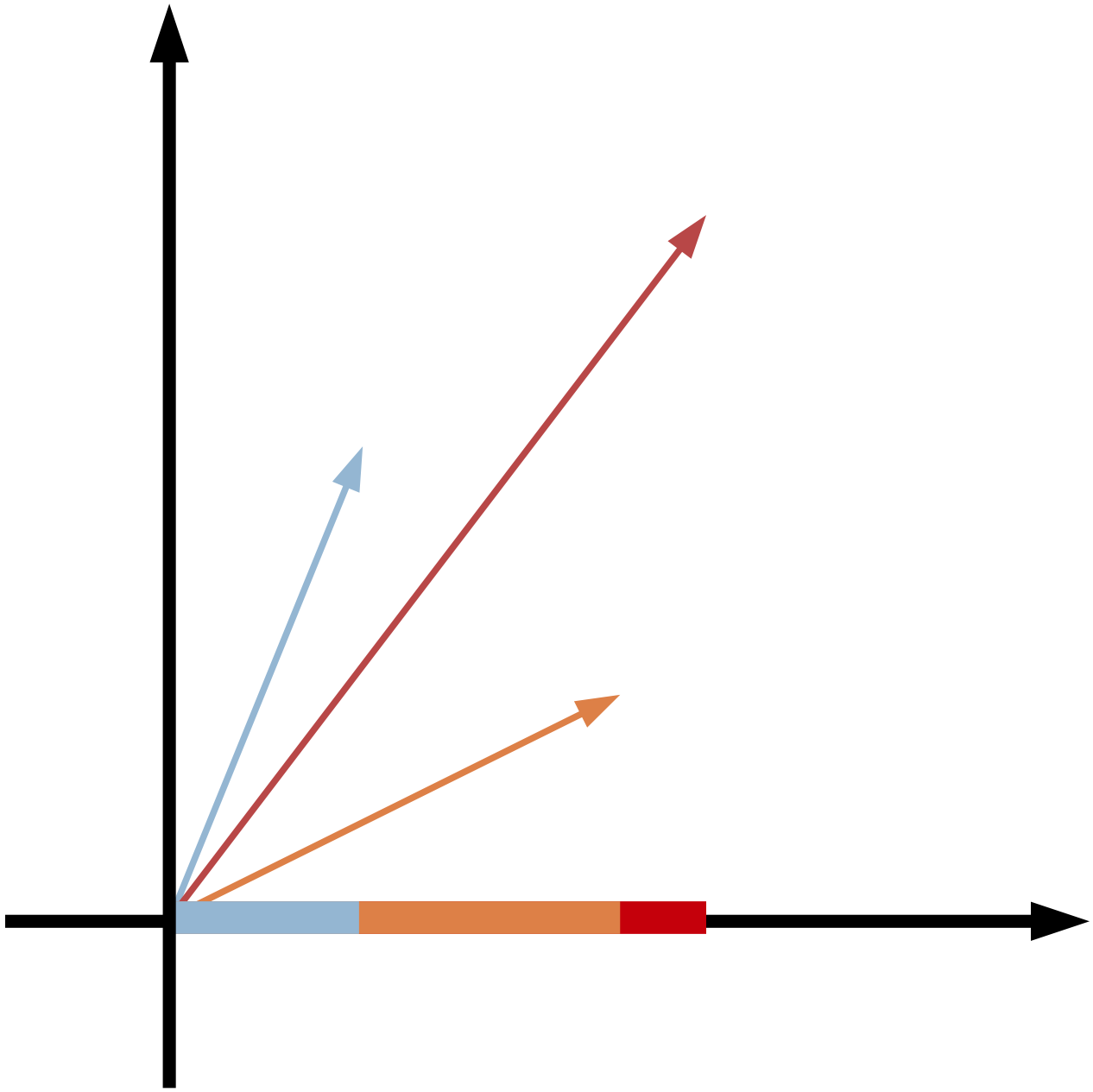




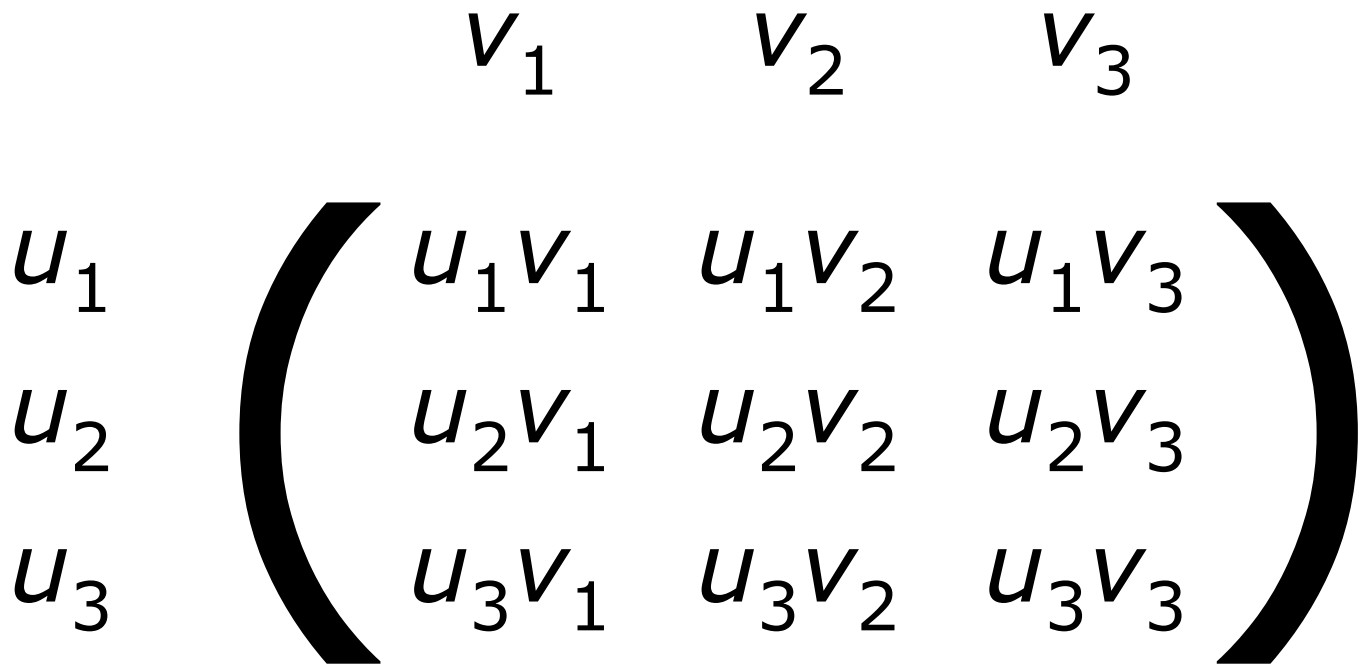


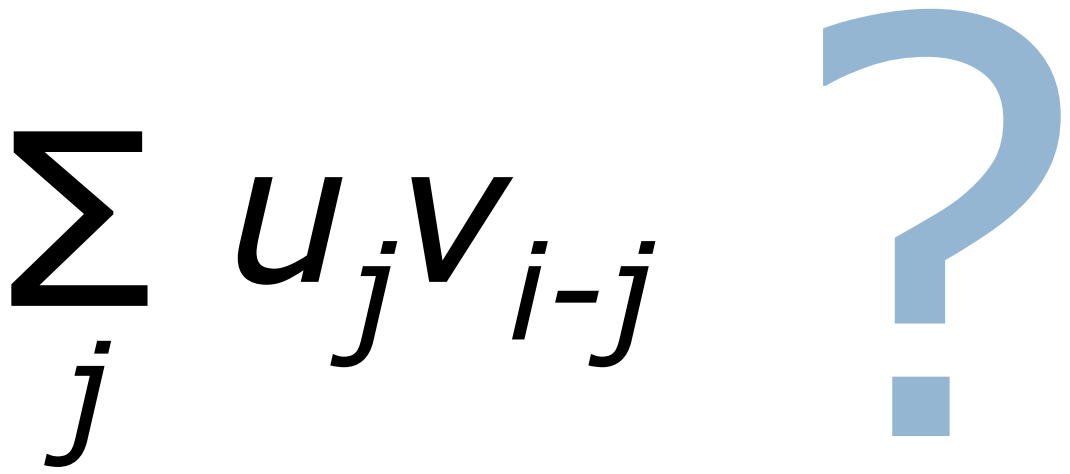






$$u_i v_j$$


v_1 v_2 v_3
u_1 u_1v_1 u_1v_2 u_1v_3
u_2 u_2v_1 u_2v_2 u_2v_3
u_3 u_3v_1 u_3v_2 u_3v_3


$\sum_j u_j v_{i-j}$
?

Composition Functions

Additive	$p_i = u_i + v_i$
Weighted Additive	$p_i = \alpha u_i + \beta v_i$
Kintsch	$p_i = u_i + v_i + n_i$
Dilation	$p_i = (\lambda - 1)(u.v)u_i + \lvert u \rvert^2 v_i$
Multiplicative	$p_i = u_i v_i$
Tensor Product	$p_{ij} = u_i v_j$
Circular Convolution	$p_i = \Sigma_j u_j v_{i-j}$

Evaluation Method

- Human similarity judgements:
  - Adjective-Noun

American country

European state

Evaluation Method

- Human similarity judgements:
  - Adjective-Noun
  - Noun-Noun

state control

town council

Evaluation Method

- Human similarity judgements:
  - Adjective-Noun
  - Noun-Noun
  - Verb-Noun

use knowledge

provide system

Evaluation Method

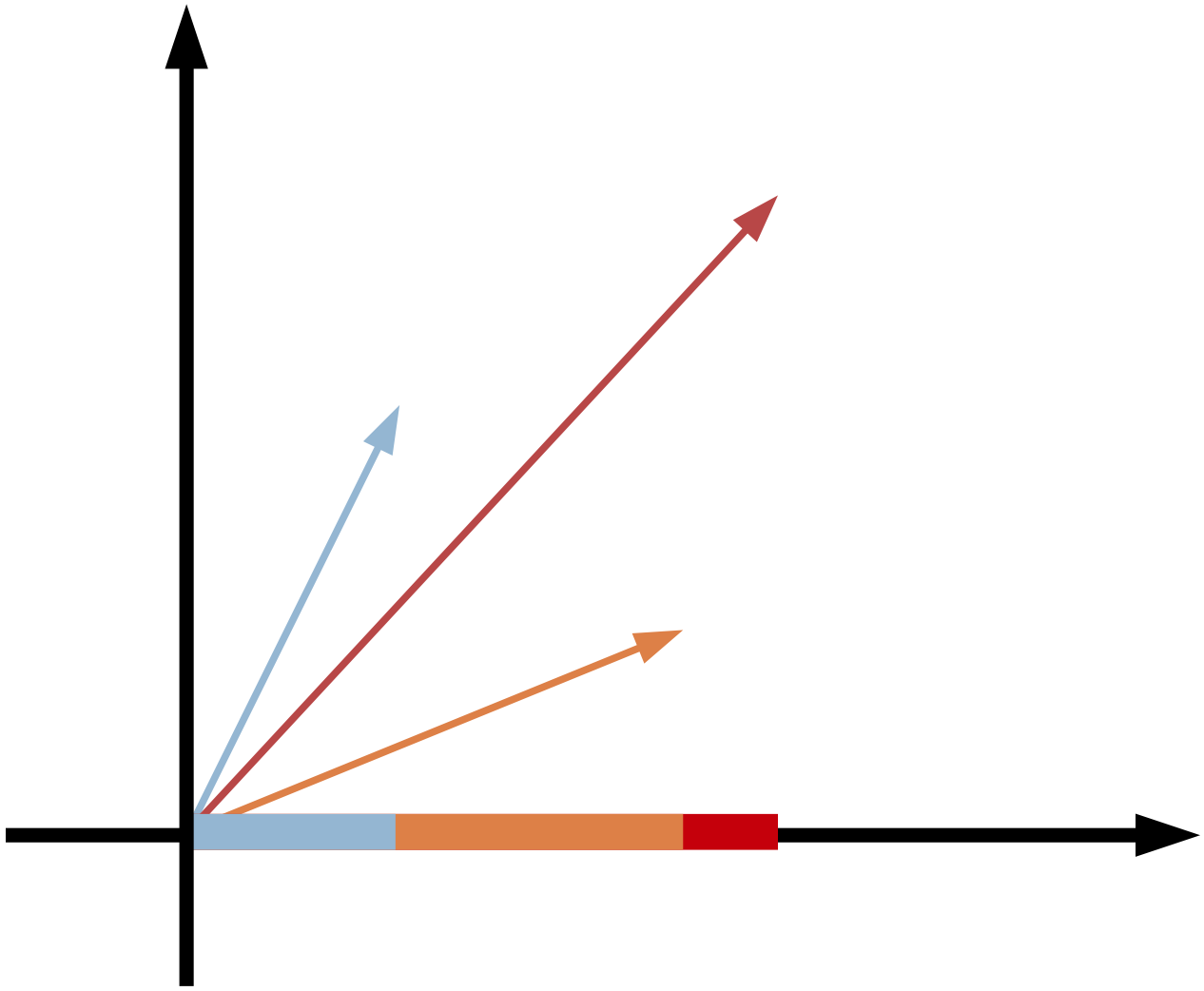
- Human similarity judgements:
  - Adjective-Noun
  - Noun-Noun
  - Verb-Noun
- Inter-subject agreement 0.52

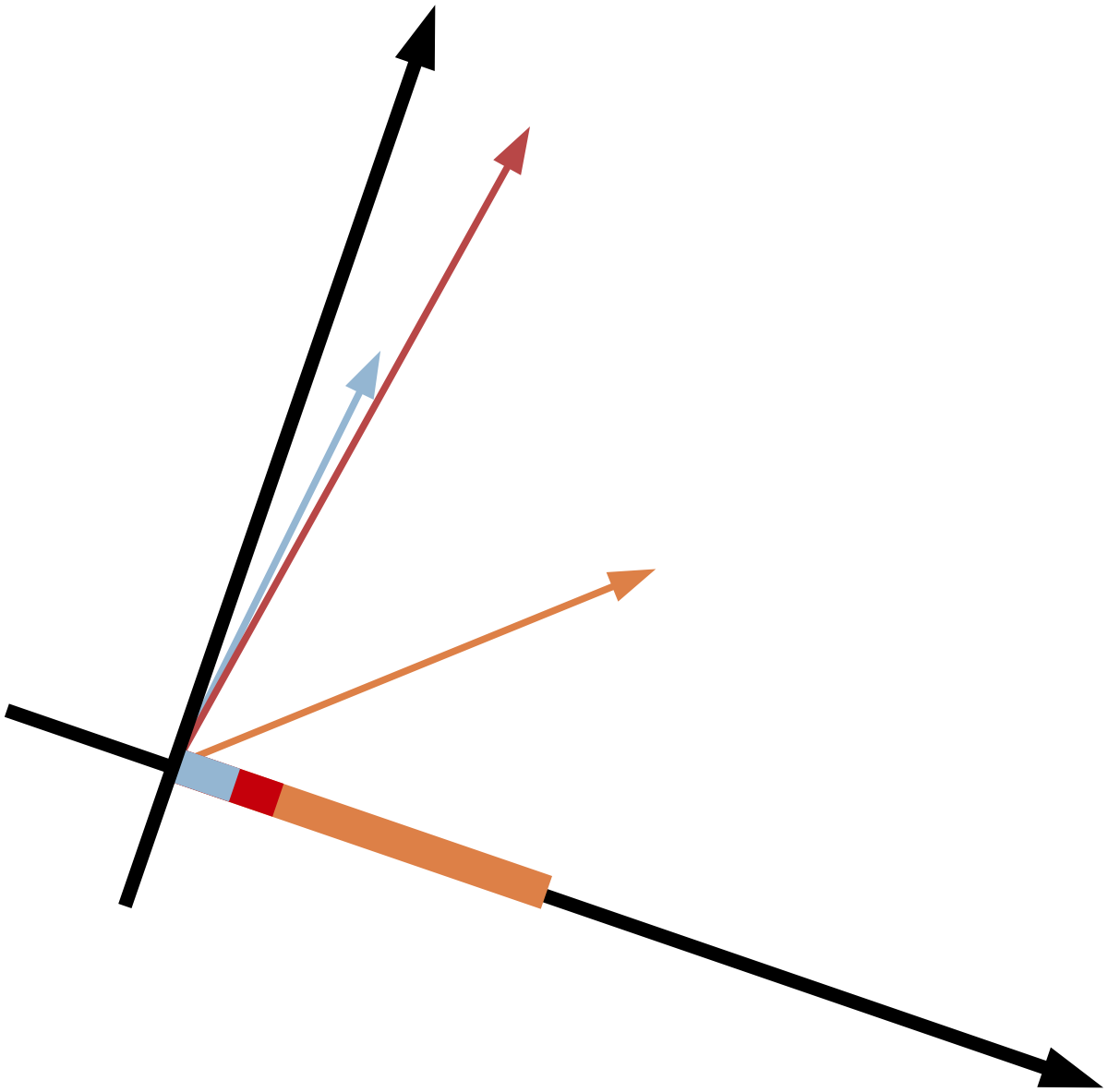
Results (window)

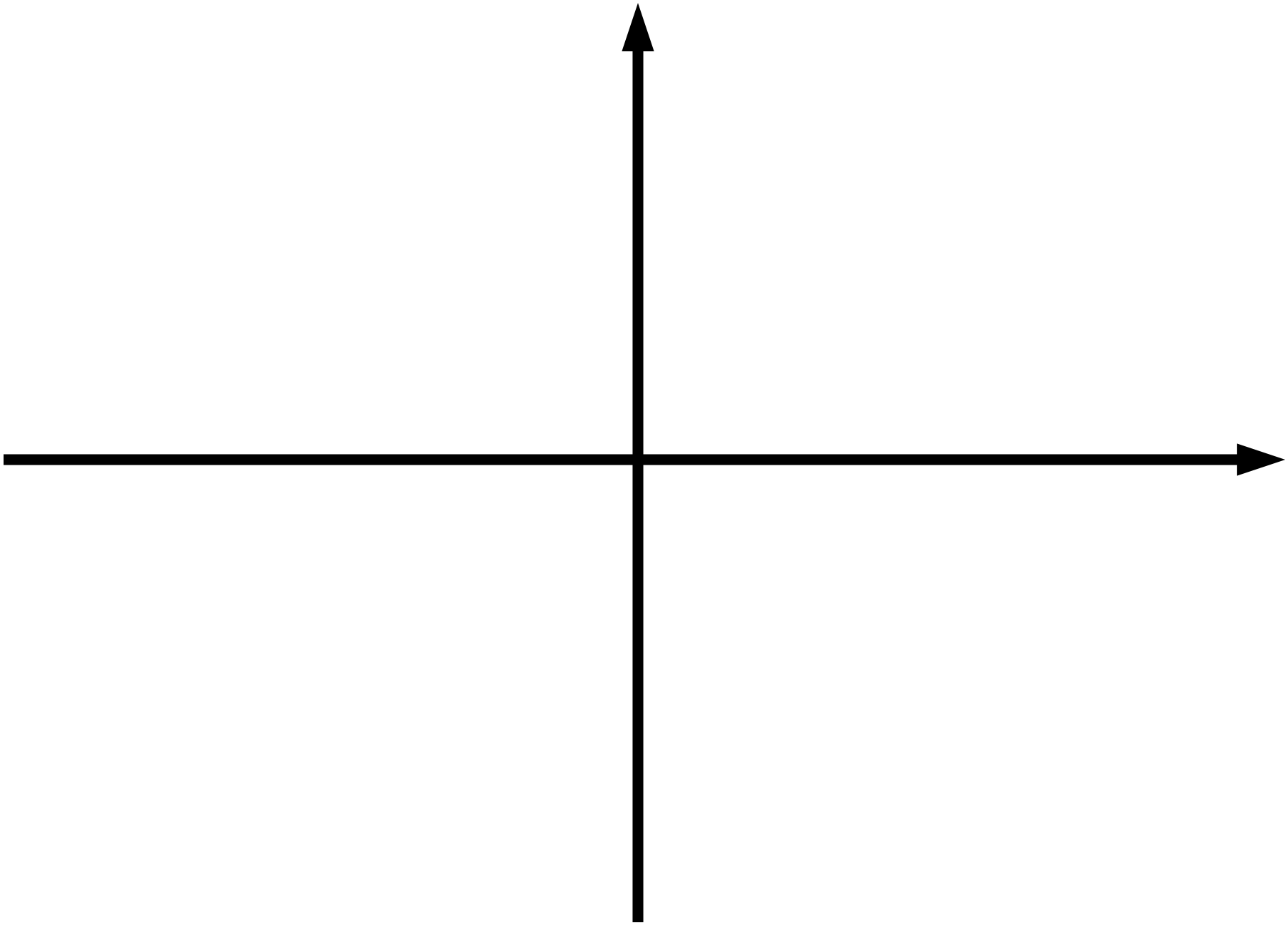
Model	Adj-N	N-N	V-Obj
Additive	0.36	0.39	0.30
Kintsch	0.32	0.22	0.29
Multiplicative	**0.46**	**0.49**	0.37
Tensor Product	0.41	0.36	0.33
Convolution	0.09	0.05	0.10
Weighted Additive	0.44	0.41	0.34
Dilation	0.44	0.41	**0.38**
Target Unit	0.43	0.34	0.29
Head Only	0.43	0.17	0.24
Human	**0.52**	**0.49**	**0.55**

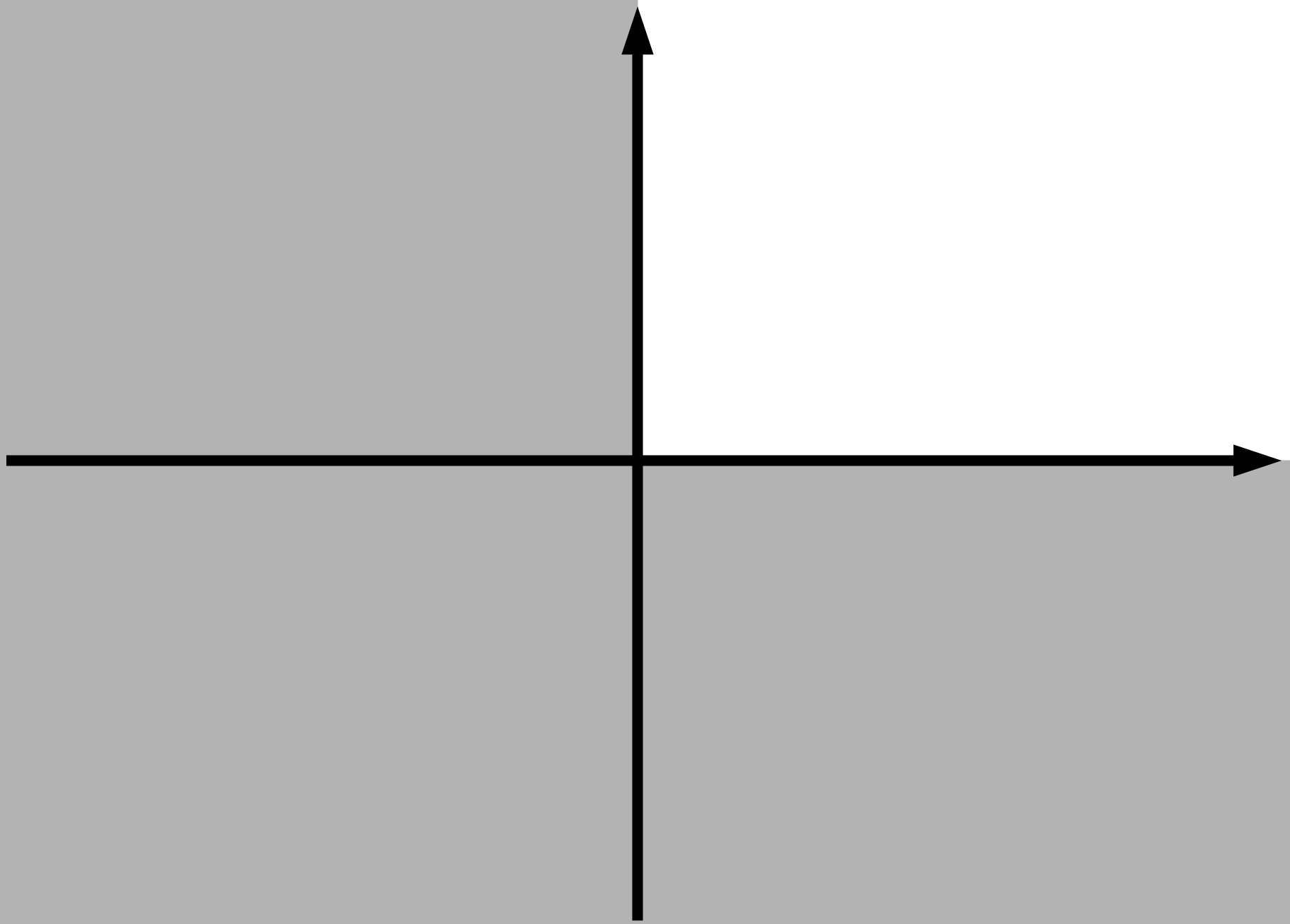
Results (document)

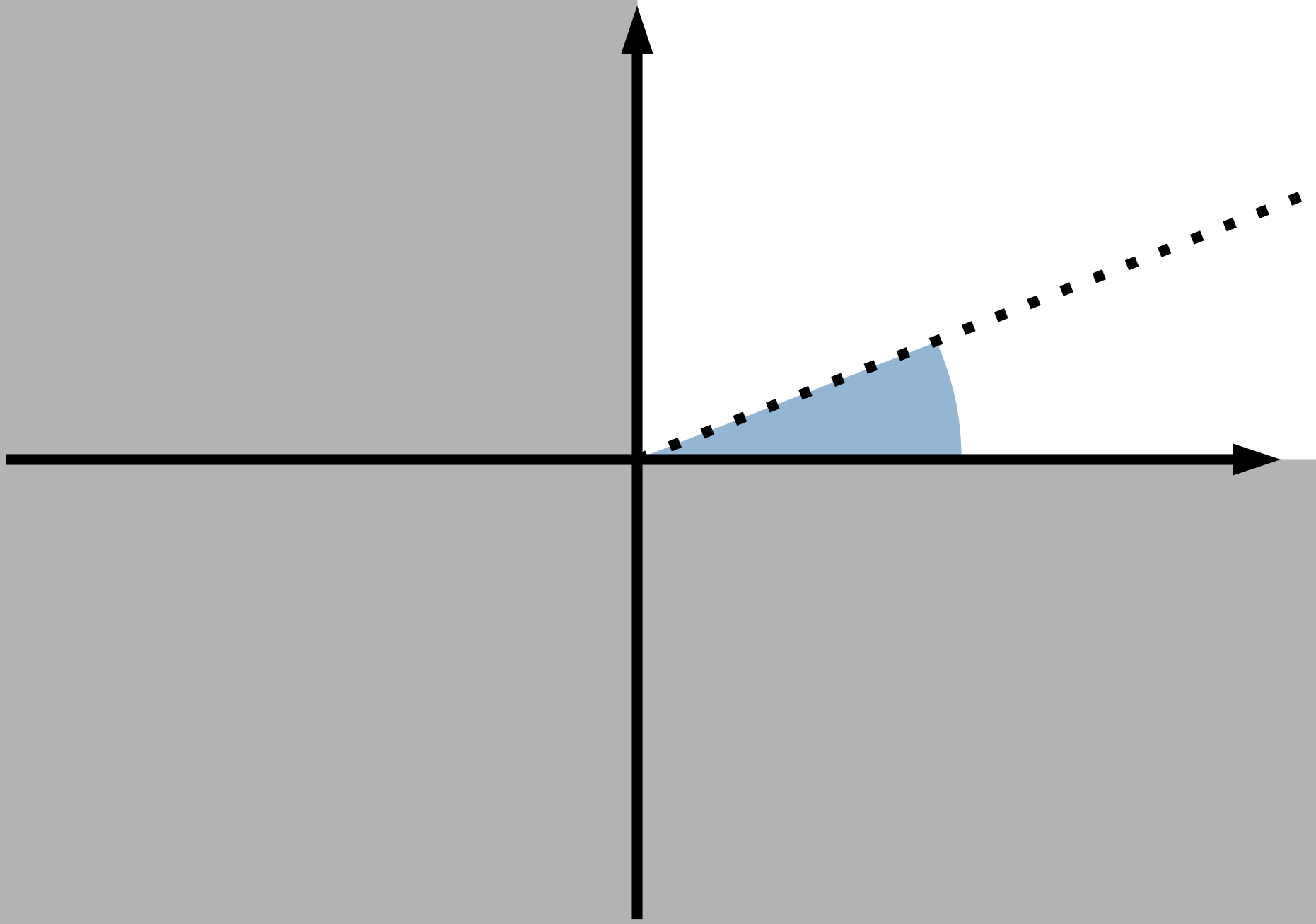
Model	Adj-N	N-N	V-Obj
Additive	0.37	0.45	0.40
Kintsch	0.30	0.28	0.33
Multiplicative	0.25	0.45	0.34
Tensor Product	**0.39**	0.43	0.33
Convolution	0.15	0.17	0.12
Weighted Additive	0.38	**0.46**	0.40
Dilation	0.38	0.45	**0.41**
Target Unit	-	-	-
Head Only	0.35	0.27	0.17
Human	**0.52**	**0.49**	**0.55**

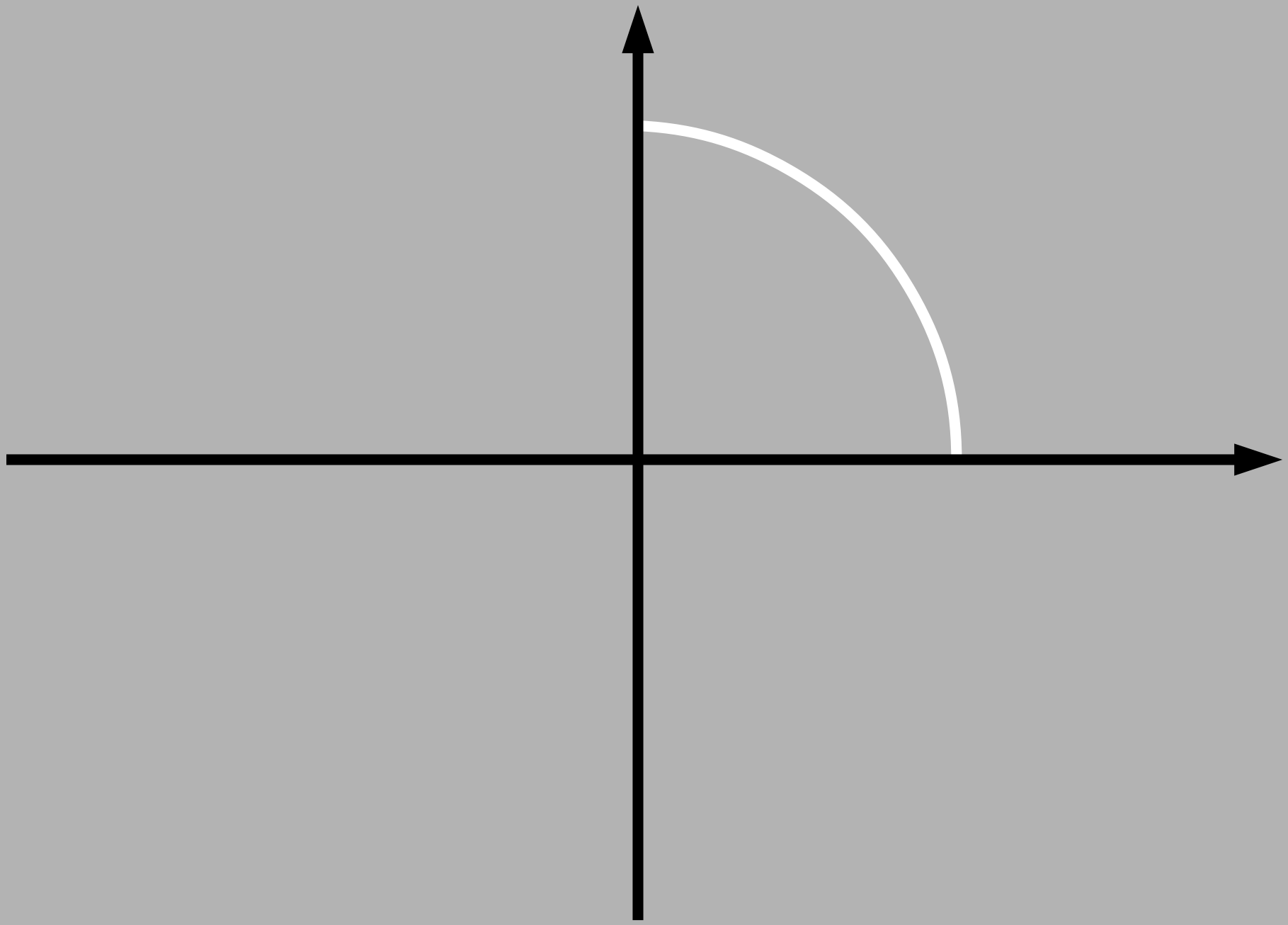












Conclusion

- Promising results from several models:
  - Multiplicative
  - (Weighted) Additive
  - Dilation
- Additional structure is needed

Notes to Slides

Slide 1: The paper I was supposed to present.

Slide 2: The surprise paper I presented.

Slide 3: The Distributional Hypothesis in the authors' own words.

Slide 4: In the 2010 paper, they consider two interpretations of “context”: either a window of 5 words either side the target word, or an entire document. When using windows, they normalise the co-occurrence frequencies according to the words' overall frequencies. When using documents, they use Latent Dirichlet Allocation as a further processing step.

Slide 5: In both cases, you can embed these semantic representations in a vector space, with the dimensions defined by context words, or latent topics, respectively. At the end of this presentation, I will return to the question of whether we really want to view these objects as vectors.

Slide 6: We can define “similarity” as the cosine of the angle between vectors.

Slide 7: The authors assume that such a semantic space is available, and do not explore different ways of constructing such a space. Their aim is to explore different ways of combining two vectors to produce a third. This is useful if we want to produce semantic representations of larger constituents.

Slides 8-9: The authors also give a more general framework, including background knowledge, but quickly abandon it. All their methods, except the tensor product, can be expressed as a function of this form.

Slide 10: All functions they consider.

Slide 11: Vector addition. This is symmetric (“commutative”), but syntax is not. The next three functions introduce an asymmetry.

Slide 12: Weighted addition. We rescale the vectors before adding them.

Slides 13-15: The Kintsch method. First, we find other vectors in the lexicon which are near to the head vector. Of these, we choose the one nearest to the modifier vector. We then add this to the sum of the head and modifier.

Slides 16-17: Dilation. We decompose the head vector as components parallel and perpendicular to the modifier. We then stretch the parallel component.

Slide 18: Componentwise multiplication. This function has no simple geometric interpretation (a point I will return to later). Each component of the result vector is the product of the components of the input vectors.

Slides 19-20: Tensor product. Unfortunately, I can't draw an interesting tensor product in two dimensions, and giving a good intuition for tensor products takes longer than a couple of minutes. However, we can express a tensor product of two vectors as a matrix, as shown in the second slide.

Slide 21: Circular Convolution. I don't think this makes sense at all, since the result depends on the *order* of the basis vectors. For instance, if we counted dogs first, then cats, we'd get a different answer than the other way round.

Slide 22: All the functions shown one more time.

Slides 23-25: The authors compare their method against human similarity judgements, given for three types of composition. In each case, human subjects had to judge the similarity of two pairs of words, on a seven point scale. A model's predictions can be compared with the average human score used Spearman's rank correlation coefficient.

Slide 26: Inter-subject agreement (average rank correlation with leave-one-out cross validation) is fairly low, which shows that this is a difficult task for humans, or that the judgements are not completely reliable.

Slides 27-28: Results for both vector spaces considered. Two baselines are provided - “target unit” calculates a distributional vector for the phrase, as if it were a single word; “head only” uses the vector for the head word.

The multiplicative method performs very well when using a window of words as a context. In particular, for noun-noun compounds, it performs as well as humans!

Verb-object collocations seem to be more difficult for these models to capture. The dilation method performs the best.

Circular convolution performs badly, as expected.

Using documents as contexts lowers performance for the multiplicative method, and for adjective-noun compounds. It raises performance in most other cases. The (weighted) additive model performs well here.

We might explain the lower performance of the multiplicative method with the LDA vector space as follows: most words will only appear in a small number of topics, so have components close to zero. When multiplying two such vectors, almost all components may then be close to zero.

Slides 29-30: Why does the multiplicative method perform so well? Before answering this, we should first ask, what exactly does the function do? One important fact to note is that it must be defined with respect to a distinguished set of basis vectors. If we change the basis, we change the result – in contrast, for addition, dilation, and tensor products, it does not matter what basis we choose. (The choice of basis also doesn't matter for more familiar multiplicative operations in linear algebra, such as the dot product and the cross product.) For this reason, the multiplicative composition function is not a natural operation on vector spaces.

Slides 31-34: But if multiplicative composition is not a natural operation, why does it work so well? Here, it is helpful to go back and look at how we constructed the space to begin with. The dimensions correspond to co-occurrence frequencies, or LDA topics. In both cases, all components are positive – in the 2-dimensional case, all vectors are in the top-right quadrant. Furthermore, if we compare vectors using cosine similarity, vectors in the same direction are indistinguishable. So, we can normalise all vectors to have length one, without affecting similarity.

Now, all we have left of the original 2-dimensional vector space is the arc in the top-right quadrant. Although these points are embedded in a vector space, they do not form a vector space on their own.

How does the multiplicative method act on this space? It pushes points to the edge (in our example, to the two end points), and it is precisely this behaviour that corresponds to our linguistic intuition that the semantic representations are becoming more specific.

So, even though the multiplicative composition function is not a natural operation on vectors, this doesn't matter because we're not really using vectors, anyway. The method works well because it exploits non-vector structure in the data. I think there would be much to gain for linguists to be explicit about this fact, and more carefully choose a structure that really is suitable for semantic representations, rather than simply applying a theory because the calculations are possible.

Slide 35: In conclusion, a number of the models show promise in modelling semantic composition. However, non-vector structure is needed – we have already seen that the multiplicative method exploits some additional structure, but much more is needed for a full theory of compositional semantics. The accompanying paper in this session (Erk and Padó, 2008) goes further, both in their "selfpref-pow" function (which is not a natural vector operation, for the same reason that the multiplicative method is not), and more explicitly in their "structured vector space" formalism.