

2025-11-25

Although it is far from being the true application, the step by step mini word example in decomposing words into components does help me a lot. Now let's take a look at the possible components offered by GPT a while ago:

1. Part-of-speech — noun/verb/adj/etc	34. Topic: politics/gov	67. Discourse connective — however/meanwhile
2. Function vs content — stopwordness	35. Temporal reference — past/present/future words	68. Topic shift — anyway/back to
3. Lemma/stem closeness — inflectional family	36. Spatiality — place/location terms	69. Contrast/emphasis — even/only
4. Negation prefix — un-, in-, non-	37. Quantity/magnitude — size/intensity words	70. Hedging — maybe/perhaps/kind-of
5. Nominalization — -ness/-tion	38. Causality — because/therefore markers	71. Disfluency/filler — uh/um
6. Past tense — -ed / irregular past	39. Modality — can/must/might	72. Dialogue/mention — @names, vocatives
7. Progressive/gerund — -ing	40. Evidentiality — "said/reported/according"	73. Quotation boundary — " / ' / ''
8. Plurality — -s/-es	41. Colors/physical properties — red/hot/cold	74. Sarcasm cue (proxy) — scare quotes, "yeah right"
9. Comparative — -er / "more"	42. Motion verbs — go/run/fly	75. Humor cue (proxy) — laughter tokens
10. Superlative — -est / "most"	43. Perception verbs — see/hear/feel	76. Narrative tense continuity — then/after
11. Gender marking — he/she/actor/actress	44. Social/kinship terms — mother/friend	77. Information status proxy — given/new hints
12. Case marking — who/whom; pronoun cases	45. Temporal expressions — dates/months	78. Anaphora potential — this/that/these
13. Diminutive/augmentative — -let/mega-	46. Sentence-start likelihood — BOS-friendly	79. Deixis — here/there/now/then
14. Agentive derivation — -er/-or	47. Phrase boundary cue — chunk edges	80. Speech act — imperative/request markers
15. Explicit negator — "not", "no"	48. Dependency head-ness — head vs dependent	81. Word-initial subword flag — <code>_</code> in SentencePiece
16. Compound boundary — subword join strength	49. Subject-likeness — nsubj proneness	82. Prefix vs suffix piece — BPE position
17. Hyphenation — x-y compounds	50. Object-likeness — dobj/pronoun obj	83. Low-frequency rarity — OOV-ish
18. Capitalization/title case — proper-noun case	51. Adjectival modifier role	84. High-frequency stopword strength
19. Numeral token — "7", "twenty"	52. Determiner-ness — the/a/this	85. Emoji class — faces/objects
20. Punctuation class — comma/period/colon	53. Preposition-ness — in/on/of	86. Hashtag/URL/email pattern
21. Concreteness — physical vs abstract	54. Clause boundary — that/which/;	87. Code identifier style — camelCase/snake_case
22. Animacy — human/animal vs inanimate	55. Coordination — and/or/but	88. Script ID — Latin/Cyrillic/Arabic/etc
23. Sentiment valence — positive--negative	56. Subordination — because/although	89. Alphanumeric mix — ID/code/serials
24. Arousal/intensity — calm--excited	57. Interrogative — who/what/why	90. Byte/garbled sequence resilience
25. Polarity scope — inside negation?	58. Auxiliary verb — do/be/have/will	91. Log frequency — unigram count prior
26. Subjectivity — opinion vs fact	59. Negation attachment — "don't", "isn't"	92. IDF-like rarity — informativeness
27. Formality/register — casual--formal	60. Scope boundary token — quantifiers/"every"	93. Mutual information with neighbors — collocation
28. Toxicity/offense cue — slur/profanity proximity	61. Relative clause introducer — who/that	94. Fixed phrasehood — n-gram idiom strength
29. Politeness — please/thanks/modals	62. Expletive subject — "there/it" (dummy)	95. Positional bias — early/late in sequence
30. Profanity marker — obscene lexicon proximity	63. Phrasal-verb particle — up/out/off	96. Line/paragraph break cue — \n patterns
31. Topic: sports	64. Light-verb pattern — "take a look"	97. BOS/EOS specialness — start/end tokens
32. Topic: finance/econ	65. Ellipsis/fragment cue — "-", "..."	98. Padding/mask behavior — training artifact
33. Topic: medicine/health	66. Coreference cue — pronoun/gender/number agree	99. Copy-ability — often copied verbatim
		100. Output bias (with tied weights) — logits lift

How to understand these possible 'axis'? Just flavors? Or something else? When the component is a generic state identifier vs a functional tag? In the above 'axis', some of them sounds very 'word specific', ie, 14, 'or', 15 'not,no', 19 'number token', 52, determiner-ness 'the/a/this', clause boundary 'that/which', 62 'there/it', 74, sarcasm cue 'yeah right'....etc. what I mean is, they don't seem to be generic word components every words have, but some of them sound like components specific for certain words. It is as if one component is just for checking if it is words like, 'not, no, and, but, which, that....'. why should these words be different directions rather than different states along a same direction?

Some words are 'functional', like 'and, or, but, however,' some words are 'state-identifier'---- which I like to call. There are words that have contents. The functional words don't have contents, they are more for supporting complex language structures. Then the contents words will have lots of nuances/flavors. I would say when our world was simple, the words should all be content-related: simply looking at the words itself, you have pretty much good idea what's the key message here. There would be no tense, no agreement matching needed. There is no concept of subject and object (though implied).

What baffled me is the 100 axis seem to be too random, just bullet points, are there any structures that can hold them together? For example, some axis are related to content, ie, topic, some are functional, ie, whether interrogative, negation attachment, relative clause introducer(?), hedging maybe/perhaps, some are intensity ie, calm-excited, some are specifically made for certain words (functions). So how to understand these axis rather than just a list?

1. Content word
 - Football
 - Tree
 - Mix
 - Blue
 - apple
2. Functional words
 - A, an, the
 - It, they, them
 - Why
 - Maybe, perhaps
 - Anyway
 - That, which

What are the biggest difference between the above two? Well, after reading content word, you have some idea, some mental picture about what could be focus. But the functional words, are to support a complex language structure, they themselves don't mean much. The functional words serve a purpose, like to 'introduce a clause', 'topic shift', 'modify adj'. Content words will bring more detailed information, like reading 'blue', I will have the color blue in my head, reading 'football', I will think of a round object played in a game. If it is a content word, it should have a core content, then on that, it can have different flavors.

I have to say most of those axis are not needed in our small word, because there is no tense, no abstractness, no complex structure, no topic (only one topic actually). They emerge to support a language system getting more complex. I would think you choose the components/ingredients so that everyone is made up of you. Similar to that you can think of all the colors made up of the same ingredients: red, blue, green. But looking at the 'axis' here, it is obvious a lot of them only apply to certain words. For example, for words 'of, then, if, it...etc', obviously, there is no gender-ness here, no animacy here, no intensity here, no topic, causality, quantity/magnitude, determiner-ness.... Actually I would say the above 99% don't apply! So it is as if we have one axis per word (type)! Is that right? What does it mean??? Why it is so?

Ok, I guess pretty unfair to say we have one axis per word. Because those special words ;the/a/this, in/on/of, that/which, and/but, uh/hum.... Those are still rare. I think most of our words are still content. GPT confirms my thought. So then we should understand those functional words have no redundancy, unlike a lot of content words, ie, amazing/fantastic/excellent, they are the same thing just with different intensity. That's why each type of functional words needs one axis.

Can I also think that, functional words play a big role in multi-head analysis? Because the multi-head analysis is really about how one word should follow another, and functional words are created for that? Here is GPT's answer.

Learning0to1

Can I also think that, functional words play a big role in multi-head analysis? Because the multi-head analysis is really about how one word should follow another, and functional words are created for that?

Short answer: **yes—function words are major anchors for many attention heads**, but they're not the only game in town.

Why they matter to multi-head attention:

- **Predictable structure cues.** Function words (the, of, not, and, if, that, who, to, in...) mark **boundaries and relations**. Heads love clean, frequent signals.
- **Linear detectability.** Their roles are easy to encode as **directions** (det-ness, prep-ness, negator-ness), so Q-K heads can spot them with simple dot products.
- **High frequency.** They appear in most sentences → lots of training signal → stable heads form around them.

Common head patterns you'd see:

- **DET → NOUN:** determiners attend forward to the head noun; nouns attend back to their DET.
- **PREP ↔ OBJECT:** "in/of/on ... [NP]" attachment heads.
- **AUX ↔ MAIN VERB:** tense/voice agreement (is/was/have/do → V).
- **NEG → VERB/ADJ:** "not/never" scoping over the predicate.
- **COORDINATION:** "and/but/or" linking parallel constituents.
- **CLAUSE BOUNDARY:** "that/which/if" marking subordinate/relative clauses.
- **WH-movement scaffolds:** wh-words linking to auxiliary/verb ("What did ... ?").
- **PUNCTUATION heads:** commas/periods signaling phrase or sentence breaks.
- **CORENFERENCE hints:** pronouns ("it/they") attending to nearby candidate antecedents.

But also:

- **Content-driven heads** exist (topic, collocation, phrase templates, "induction heads" copying repeated names/keys).
- Later layers blend **syntax + semantics**; some heads key off **content words** (e.g., *protein* → *gene*, *basketball* → *NBA*).

So your mental model is good: **function words ≈ scaffolding for "who follows/depends on whom,"** giving multi-head attention clear hooks for syntax and scope; **content words** supply the semantic payload that other heads specialize in. Both are needed, but function words punch above their weight for structural heads.

What does it mean by 'linear detectability'? Take a common head example, 'neg→verb/adj', since it comes from linearly mixing the meaning components, what components could get mixed here'? That's token dependent.

I am still trying to understand why a query/token could be formed by linearly mixing the meaning components. Take word 'not' as an example.

If it is word 'not', then if we are analyzing from the perspective of 'negative-verb or negative adj', then its query is a verb or adj. Its key is a 'negation tag?' We need to understand prior to arriving at the conclusion that "not's query is a verb or adj' from the perspective of 'negative-verb/adj', why you need to linearly combining meaning components to form Q and V? What meaning components might need to be picked so to form such query? The 'negation', 'explicit negator' one for sure. Then verb-ness, adj-ness? We've seen how so many other 'axis'/directions are non-relevant, so I am suspecting during this linear mixing step, a lot of irrelevant axis will be downplayed, while the relevant one will be highlighted.

From Det-noun perspective, it asks nothing, providing nothing? Almost from other heads' perspective, no query, no key! Step back.

2025-11-26

What does negation head learn? GPT's answer is full of jargons. Maybe we should ask: what is negation head about? Why the query for 'not' should be predicates? And what is predicates? "not well", is well a predicate? Well, turns out I am too unfamiliar with grammatical talks, predicate is strange to me. So ask GPT for clarification.

Possible way to finish 'not':

- **Main/lexical verb:** *eat, go, allow, rain*
"not go", "did not eat", "not allowed"
- **Verb phrase with auxiliaries:** *is/was/has/can + V*
"is not working", "did not see"
- **Predicative adjective:** *be + adj*
"is not happy", "was not possible"
- **Predicative noun (nominal):** *be + noun*
"is not a problem", "was not the cause"

•
So let's recap the big picture. I feel like I am suffocating beneath the ocean of details. There are two main types of words, function words that do have a lot of meanings, and content words that convey information and meaning. And then words get together, certain words get along, certain don't. So multi-head analysis is focusing that. Why? Because we are in the business of predicting the next words! And what basis? Basis on how certain words would get along and certain don't.

Then what are the ways that words do get along then?

Functional words play a big role, because if they don't have meanings themselves, then why need them? Obviously they are there to support the structure, to connect words! But as GPT pointed out, content-driven heads are also part of the picture. Words can get together because of their contents: they fall into the same topic, collocation (like a collection of similar words), phrase template (what is it?) and induction head (what is it, do diagnose certain repeated pattern like AB, ... AB????)? Anyway, long story short, some heads are functional words specific, some heads are content driven.

Now think about functional words first, in terms of how they define the rules for the neighboring words.

1. Determiner, like, a, an, the --, a has to follow a singular noun, an as well, but with a vowel starter.
2. Preposition ->object, words like 'in/of/on', what kind of words must follow? What kind of rules?
3. Auxiliary verb like, 'be, do, have', what words to follow and what rules to comply? "do remember", 'do forget', ok, a verb is needed, tense must match. And GPT says 'voice agreement', what is that?
4. Negation-verb/adj, words like 'not'. 'Not well', 'not welcome', (do) not eat. Ok, verb and adj can follow, rules?

5. Coordination: words like 'and/but/or', they are linking parallel constituents. *What to follow and what rules?*
6. *Clause boundary: words like 'that/which/if'.*
7.

Ok, you know what's strange, these heads seem to be 'words specific', like, if there is word 'no/not', then the negation head, if the words have 'and/but/or', then the coordination head. Then you would expect lot of special treatment for such analysis, such as if branching "if the words are and/but/or do this", if the words are no/not then do this..., so lots of special treatment and hence discontinuities. But it's not. From what I am seeing, all you need is 16 different sets of W matrices to produce the corresponding K, Q and V. Is it time to ask GPT about how you actually control which head is doing what? I guess the answer is as usual: the system learns itself by putting in the right parameters into the matrices. So maybe we should hold on this question.

But since we know it is just different parameters in the matrices picking out different meaning ingredients, let's imagine what could have happened. For the negation head analysis. First of all, it needs to mix the meaning components to create the right query and key for each token. What should it focus on? Maybe the first question is: what does it need to accomplish? Then what does it need in order to accomplish that? The answer to the second question should be related to the raw meaning ingredients, because after all, we are mixing the meaning components to form key, query and value!

So questions:

1. For negation head analysis, what does it need to accomplish?
2. What does it need in order to accomplish that goal?

Short answer to first question: it needs to create the right K, Q, V for the right words. Then let's think about the words that are relevant, and then the irrelevant. For the relevant words, they would be 'no, not, adj, verb'. Then they no/not will have query asking for adj, verb, predicate, with the right tense and tone (maybe?). On the other hand, they will advise their role as 'negator, negation'. So key and query are pretty high level emphasizing their main role, ~~then the value would be more specific, pinpointing the word?~~ Hold off on values yet. Now knowing what they need to produce, let's focus on need, what does it need in order to achieve that.

For the word "not", what does a negation head learn?

Think of a small set of latent directions:

- u_{negator} : explicit-negator cue ("not/never/no").
- u_{pred} : predicate-likeness (verbs, predicate adjectives).
- u_{sent} : sentiment/polarity carrier (good/bad, happy/sad).
- u_{scope} : scope openness (no clause barrier / short rightward distance).
- u_{barrier} : clause/coordination boundary (comma, "but", "that").
- Position (from RoPE): short rightward distance bias.

First of all, these directions are in what space? Meaning space? I assumed so. So it means, the meaning space should tell us if a word is a negator, predicate-ness, sentiment, ok easy to understand. But not sure about the scope and barrier. I mean, how could you tell the scope and barrier from a single word? So we should go back to the 100 axis: is scope and barrier here? My suspicion is, this is from some functional word, because a content word is about the information itself, not to be supporting a language structure. Turns out I am right. 60. Scope boundary token -quantifiers 'every'. Ah ha!

I think we can say, sometimes, one word is for one dimension (direction!)

2025-11-28

i mean i can imagine negation head react strongly to token of 'not', but i can't imagine it's reacting to token like 'basketball'

Exactly. A "negation head" will **hardly react** to "basketball".

Why:

- Each head's scores are $q_i^\top k_j$. Training makes W_Q, W_K so that:
 - For "not": q_{not} has big components along **predicate-seeking** directions.
 - For **predicates** (verbs/ predicative adjectives): k_{pred} aligns with those directions.
 - For **content nouns** like "basketball": $q_{\text{basketball}}$ has **tiny** projection on those directions → **small dot products** → **low attention weights** in that head.

So, in "They did **not** play **basketball**":

- **Negation head**: "not" → **play** (strong). "not" → "basketball" (weak/none).
- A different **object/linking head** handles **play** → **basketball**.

Bottom line: heads run everywhere, but only **tokens with the right features in the right context** (e.g., "not") produce queries/keys that **light up** that head; unrelated tokens (e.g., "basketball") look almost invisible to it.

I think GPT misunderstood my question. My question is about the query and key itself, while GPT is talking about the end result of QK^T . I was wondering what the key and query would look like for an irrelevant word 'basketball' under the negation head analysis. But GPT seems to be not caring so much about the intermediate Q, K, instead it says, the end result of the dot product will tell us, low attention

weights. So a head is analyzing what a token's need and offer from a certain perspective. So far I was concerned with a relevant word. But what if the word is not relevant at all in this analysis? I was curious about what kind of key it can provide and what kind of need it has? But seems that they don't matter, because the end result is for the dot product to be small so that the token does not match anyone's need in this analysis. Ok. But think about this, if the token is irrelevant, how can you make sure the dot product of its Key with other Query will always be small? Similarly, how can you make sure the dot product of its Query with other Key will always be small? So I push GPT, seems that, for irrelevant tokens, the key and query will be close to 0! Only in this case, the dot product of them with anything else will be close to 0, hence 'small'. Hmm, but something else I noticed during my dot product exercise: what will happen when the dot product is 0, versus negative? Then you should understand the meanings behind: 0 means irrelevant: key is irrelevant to what the query is asking. But negative means something else: they are relevant, but just opposite of what you want. So in the softmax scheme, dot product of close to 0 will be small, hence unlikely a match, but dot product of negative number will be even smaller! Because it's the opposite of what you want! Is that right? So the softmax is punishing an opposite match even more!