The linear order of elements in prominent linguistic sequences: Deriving Tns-Asp-Mood and Greenberg's Universal 20 with n-grams

Stela MANOVA
stela.manova@univie.ac.at

Current NLP research uses neither linguistically annotated corpora nor the traditional pipeline of linguistic modules, which raises questions about the future of linguistics. Linguists who have tried to crack the secrets of deep learning NLP models, including *BERT* (a bidirectional transformer-based ML technique employed for Google Search), have had as their ultimate goal to show that deep nets make linguistic generalizations. I decided for an alternative approach. To check whether it is possible to process natural language without grammar, I developed a very simple model, the *End-to-end N-Gram Model (EteNGraM)*, that elaborates on the standard *n-gram* model. *EteNGraM*, at a very basic level, imitates current NLP research by handling semantic relations without semantics. Like in NLP, I pre-trained the model with the orders of the TAM markers in the verbal domain, fine-tuned it, and then applied it for derivation of Greenberg's Universal 20 and its exceptions in the nominal domain. Although *EteNGraM* is ridiculously simple and uses only bigrams and trigrams, it successfully derives the attested and unattested patterns in Cinque (2005) "Deriving Greenberg's Universal 20 and Its Exceptions", *Linguistic Inquiry 36*, and Cinque (2014) "Again on Tense, Aspect, Mood Morpheme Order and the "Mirror Principle"." In *Functional Structure from Top to Toe: The Cartography of Syntactic Structures 9*. *EteNGraM* also makes fine-grained predictions about preferred and dispreferred patterns across languages and reveals novel aspects of the organization of the verbal and nominal domain. To explain the *EteNGraM*'s highly efficient performance, I address issues such as: complexity of data versus complexity of analysis; structure building by linear sequences of elements and by hierarchical syntactic trees; and how linguists can contribute to NLP research.*

1. Deep learning, n-grams and the future of linguistics

Theoretical linguistics and current NLP models differ in the way they approach the form-meaning relation in language. I will explain the issue based on Manova et al. (2020) who define three types of approaches to the form-meaning relationship:

(i) the relationship is non-directional, i.e. form and meaning emerge simultaneously (pairings of form and meaning such as *-s* <PL> are typical of this type of approaches);

(ii) the relationship is directional and from-meaning-to-form (e.g. when the feature <PL> is associated with a terminal node on a syntactic tree (Distributed Morphology, Cartography, Nanosyntax) or with a cell or a set of cells in a paradigm-based approach (Paradigm Function Morphology); and

(iii) the relationship is directional and from-form-to-meaning (e.g. a machine or a human identifies *-s* and the association with meaning is postponed, or does not happen at all).

Theoretical linguistics approaches are, as a rule, of (i), (ii) or a mixed (i) & (ii) type, while current NLP approaches are of type (iii). This leads to significant differences in the analyses and leaves the impression that current NLP does not have anything in common with theoretical linguistics. This impression is further enhanced by the fact that NLP has always been entirely focused on application and efficiency while theoretical linguistics has never been. Additionally, in the past decade a paradigm shift has happened in NLP and linguistically annotated corpora and the traditional pipeline of linguistic modules have been substituted by raw data from the web and deep neural networks (DNNs), respectively. There are different types of DNNs and things are fairly complicated to be explained in detail here; for a linguistic perspective

* I wish to thank Professor Guglielmo Cinque for his insightful research on the topic, without his research my work would not have been possible.

I dedicate this research to all linguists born in poor countries and territories, with a small number of native speakers, who deal with prejudices and discrimination in linguistics.

on the issue, the reader is referred to Linzen & Baroni (2021). The core facts: DNNs are typically "end-to-end", i.e. a deep network is directly trained to associate input (e.g., text in one language) to a corresponding output (e.g., text in another language). An alternative way of looking at what it means to be "end-to-end" would be: Trying to find a route for problem solving by knowing the problem's solution in advance. For example, we take a text in, let us say, Russian (input) and the same text translated into, let us say, German (output) and the task is translation from Russian into German. In this case, the solution of the problem is the German translation. With the help of the German translation, the DNN will learn which sequences of words in Russian correspond to which sequences of words in German. Deep learning takes place in the hidden layers of the network and from input to output, each hidden layer handles increasingly complex information. It is a non-trivial task to crack the way of "thinking" of a DNN because all relations between the the layers, including recurrent activation (if the network is recurrent), are allowed, i.e. the network alone "decides" which route to activate (and similar to human DNNs (brains) with different IQs, different DNNs come to a solution of the same problem through different routes). Returning to the Russian-German translation, what the network has learned, roughly the statistics of the corresponding Russian-German form sequences, can be used for translations of unknown texts from Russian into German and vice versa. The more we train the network (the more parallel translations we give to it), the more precise (human-like) the translation of unknown texts (compare with the human brain: the more a human practices German-Russian translation, the better s/he gets), which is the explanation for why e.g. Google Translate that relies on a DNN performs better when translating from and into languages for which there are many texts on the web and produces less convincing translations from and into less popular languages.

Although the general logic of DNNs is easy to grasp, the programming of a DNN and the linguistic monitoring of its work are difficult tasks.[1] For example, BERT (Bidirectional Encoder Representations from Transformers), a machine learning technique currently employed in Google Search "is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications." (Devlin et al. 2018: 4171). In other words, BERT involves two steps: pre-training and fine-tuning. Additionally, BERT is known in two model sizes: $\text{BERT}_{\text{BASE}}$ (L=12, H=768, A=12, Total Parameters=110M) $\text{BERT}_{\text{LARGE}}$ (L=24, H=1024, A=16, Total Parameters=340M); "L" stands for the number of layers (transformer blocks); "H" stands for hidden size, i.e the number of hidden neurons; and "A" stands for the number of self-attention heads, i.e. heads that attend to the previous hidden states. BERT works with sentences but "sentence" is defined as an arbitrary span of contiguous text, i.e. BERT sentences are not always actual linguistic sentences. The input token sequence to BERT may be a single sentence or two sentences packed together.

Regarding the linguistic interpretation of a DNN performance, McCoy et al. (2020) check whether if a neural network architecture is tested multiple times on the same dataset, it will make the same or, at least similar, linguistic generalizations each time. They establish that the answer to this question depends on the testing set: if the testing set contains distributions like those in the

[1] For a hands-on experience with DNNs, the curious reader can visit the website A Neural Network Playground, part of Google's TensorFlow educational resources: www.tensorflow.org. Popular-science explanations of how to understand the Playground at: https://blogs.scientificamerican.com/sa-visual/unveiling-the-hidden-layers-of-deep-learning/.

training set, the DNN makes (almost) the same generalizations; if the testing set includes previously unseen distributions, the DNN tends to take different routes each time. The authors are surprised by these results but the DNN behavior seems completely understandable: Imagine that a human being (a human brain is a neural net) should go through an unknown labyrinth many times, most probably, each time s/he would try different routes.

DNNs and similar developments in recent NLP research have raised questions about the future of linguistics; and, consequently, on the linguistic side, proposals for possible interactions between linguists and the NLP community have been articulated, see e.g. Baroni (2021). I am, however, afraid that what Baroni (2021) and Linzen & Baroni (2021) suggest as a solution to the problem would work only for a handful of linguists, mostly those with (advanced) degrees in computer science (CS), electrical engineering, mathematics, and the like:

> On the one hand, linguists' know-how in probing grammatical knowledge can help develop the next generation of language-processing DNNs… On the other, studying what the best DNNs learn about grammar, and how they do so, can offer new insights about the nature of language and, ultimately, what is genuinely unique about the human species. For this line of work to be effective, linguists will need to be closely involved in developing relevant network architectures, training them on appropriate data, and conducting experiments that address linguists' theoretical concerns.
>
> Linzen & Baroni (2021: 15)

I think that in order to participate in a fruitful dialogue with the NLP community, linguists need to see the advantages of a simple and easily-applicable NLP model over traditional linguistic analyses, including what it means to model semantics without semantics, which is the case in current NLP research. To be convincing, the illustrating model should also address issues that have been acknowledged as important by people of different theoretical persuasions. Note, however, that what is considered important in linguistics, e.g. linguistic universals, is not necessarily of interest to the NLP community. Roughly, because we do not speak with universals, NLP does not need them. Nevertheless, as we will see later, linguistic universals can reveal important facts about possible and impossible linear patterns of elements in language, which is highly relevant to NLP research. Therefore, in the present study, I propose, pre-train and fine-tune the End-to-end N-Gram Model (*EteNGraM*), a very simple NLP model that elaborates on the standard n-gram model (Jurafsky & Martin 2020, Chapter 3).

A n-gram is a contiguous sequence of n-elements. A single element (e.g. phoneme, morpheme, word) is a unigram, two elements form a bigram, a sequence of three elements is a trigram, and so on. N-grams are simple structures and consequently easy to program and their computation does not require much memory. Such features make n-grams an attractive and robust solution to some NLP problems and also explain why the n-gram model has not been completely outranked by DNNs. So far, n-grams have been used in applications such as sms writing, spelling correction and grammar checking, online shopping, generation of automatic email responses, etc. (cf. Jurafsky & Martin 2020). By contrast, DNNs are complex models, a profound knowledge of CS is necessary to design and program a DNN, DNNs also require powerful hardware with a solid amount of operative memory, which will certainly turn into a problem for ML algorithms in the future because the amount of web data also grows rapidly. Devlin et al. (2018) confess some similar problems with BERT's architecture, namely at some point the model becomes so big that it is impossible to handle.

Regarding whether n-grams relate to DNNs, one can think of a DNN analysis of language data as operating with n-grams of different types and different orders simultaneously, cf. e.g. the

explanation of Google's Neural Machine Translation system in Wu et al. (2016) who speak of characters, words and sub-word units ("wordpieces").

In NLP, it is well-known that a given text (a closed set of elements) can be reproduced based on the n-grams it contains; and that higher-order n-grams ensure a more coherent reproduction. This has been demonstrated with the corpus of Shakespeare's works, i.e. with the four-grams of the Shakespeare corpus, a text written by a machine sounds a lot like Shakespeare (Jurafsky and Martin 2020, Chapter 3).

It should also be mentioned that bigrams are not something new to theoretical linguistics: Ryan (2010) models variable affix order in Tagalog with the help of bigrams; naive discriminative learning (Baayen et al. 2011) operates with bigrams of letters; Mansfield et al. (2020) analyze "free" prefix order in Chintang in terms of bigrams; semantic scope in Athabaskan has been explained in terms of relations between two affixes (Rice 2001); sequences of two suffixes play an important role in the organization of the mental lexicon (Manova & Knell 2021 with data from English); see also the discussion in Manova (to appear) on how restrictions on the order of affixes in general have been tackled in the literature so far, namely as sequences of primarily two affixes but without referring to such sequences as bigrams. Syntactic phrases, VP, NP, and so on are not bigrams, strictly speaking, because the two elements of a bigram must be neighbours in a linear sequence of elements, which is not always the case in syntactic phrases.

The remainder of the paper is organized as follows. In section 2, the complexity issue both in mathematics / CS and in linguistics is addressed and it is shown that the only objective way of measuring complexity is by assessing the complexity of the solution (analysis). In section 3, the *EteNGraM* model is introduced. In section 4, *EteNGraM* is first pre-trained with the existing and non-existing TAM orders from Cinque (2014); the model is then fine-tuned and employed for derivation of Universal 20 and its exceptions. Section 5 tackles the fine-tuning procedure and there is also a discussion on what linguists should learn from *EteNGraM* in order to participate in a fruitful dialogue with the NLP community. *EteNGram*'s uniform derivation of the TAM and Universal 20 orders is compared with Cinque's derivations of these two prominent sequences in section 6, as well as with alternative derivations of Universal 20 from the literature. Afterwards, in section 7, *EteNGraM* is challenged with additional data for Universal 20. In section 8, *EteNGraM* demonstrates its full power by making fine-grained predictions about preferred and dispreferred patterns of Universal 20 cross-linguistically. Section 9 concludes the paper.

2. Complexity as a property of analysis

Both language and mathematics are beautifully-organized systems. In mathematics, beauty always means simplicity: a problem may have more than one solution but the simplest solution is the most valuable. For example, the task of summing up the numbers from 1 to 100 has at least two solutions: 1+2+3, and so on to 100; and a more elegant one based on the observation made by the young Gauss that 100+1 = 99+2 = 98+3, and so on to 51+50, which means (1+100)*50=5050. Both solutions give the same result, 5050, but the first one is uninteresting, while Gauss' solution has been used as a formula for the sum of an arithmetic progression ever since. In other words, mathematicians understand the simplicity-complexity issue as a property of the solution (analysis), while in linguistics the common belief is that complexity is a property

of the data (language). This does not mean that in mathematics there are no easy problems (data) and difficult problems (data), just the solution can turn a difficult problem into an easy one and therefore the only objective way of measuring complexity is through the assessment of the complexity of the solution. Clearly, the same holds for CS where there is a special system, the so-called *Big O*, for measuring the complexity of a solution. In this paper, among other things, our goal is to show that language is indeed as beautifully organized as mathematics and, consequently, language's complexity is like that in mathematics--depends on the solution.

Our data and the information about attested and unattested patterns come primarily from Cinque (2005, 2013, 2014). And it must be made explicit here that without Cinque's research, this study would not have been possible.

Finally, the research reported here is mathematical in nature and neither revises nor formulates a linguistic theory but approaches a problem with the goal to solve it, whatever the consequence for any theory.

3. Introducing *EteNGraM*

My goal at this stage is to keep *EteNGraM* as simple as possible (recall what was said about simplicity in the previous section). The different steps of the algorithm are in capital letters. The four numbers "1", "2", "3", "4" (unordered sets of them) are the input, i.e. these numbers will represent both all TAM orders in the verbal domain and all Universal 20 orders in the nominal domain. (I use four numbers because the sequences I am going to derive have four elements each: V-Asp-Tns-Mood and N-A-Num-Dem.) The ordered set, 1-2-3-4 (ascending order), will serve as output and I will refer to it as the default pattern.

A. *Derivation by n-grams*

The default pattern is the most frequent pattern when all pattern's elements are overt. The default pattern serves for derivation of existing patterns in terms of bigrams and trigrams, as illustrated below.

B. *Deriving the default pattern with bigrams and trigrams*

B1. Bigrams and trigrams based on immediate neighborhood

(1) Bigrams of neighbour elements

1-2-3-4 (default pattern)
1-2
2-3
3-4

(2) Trigrams of neighbor elements

1-2-3-4 (default pattern)
1-2-3
2-3-4

B2. Bigrams and trigrams of non-neighbor elements

The classical n-gram model does not have this type of bigrams and trigrams. I, however, postulate them since it is not always the case that all elements of a linguistic sequence are on the same side of the base or overt.

(3) Bigrams of non-neighbor elements

1-2-3-4 (default pattern)
1 3 → 1-3
1 4 → 1-4
2 4 → 2-4

(4) Trigrams of non-neighbor elements

1-2-3-4 (default pattern)
1 3-4 → 1-3-4
1-2 4 → 1-2-4

Since all bigrams and trigrams in this subsection are derived left-to-right, at this stage *EteNGraM* is a unidirectional left-to-right model. (For comparison, BERT is bidirectional.)

C. *Pre-positioning, post-positions, and stranding*

It is well-known that languages of the world tend to be either prefixing (prepositional) or suffixing (post-positional) but that not all elements always either precede or follow the base. Obviously, the default sequence 1-2-3-4 consists of a base (V in the verbal domain and N in the nominal domain) and three non-base elements. If at least two of the non-base elements follow the base, the pattern will be classified as post-positional. From now on, all **bases** will be **bolded**. Post-positioning is illustrated in (5). Likewise, if at least two elements precede the base, the pattern is a case of pre-positioning (6). The single elements that either follow or precede the base are 'stranded' and from now on they will be colored in green.

(5) Post-positioning with stranding

1-2-3-4 (default, base under **1**)
4-**1**-2-3
3-**1**-2-4
2-**1**-3-4

(6) Pre-positioning with stranding

1-2-3-**4** (default, base under **4**)
1-2-**4**-3
1-3-**4**-2
2-3-**4**-1

In (5) and (6), the bigrams formed by the base and the stranded element violate the rules postulated in B but there is no way to repair these bigrams: a stranded element is trapped and cannot move. Stranded elements are therefore excluded from the n-gram analysis. Note also that although the bigrams consisting of 'base + stranded element' in either order are not in accord with the rules in B, the rest of the bigrams and trigrams all are well-formed. For example, in (5),

we have 1-2-3 (with stranding of 4), 1-2-4 (with stranding of 3), and 1-3-4 (with stranding of 2). Likewise for (6): 1-2-4 (with stranding of 3), 1-3-4 (with stranding of 2), and 2-3-4 (with stranding of 1).

Now, since stranded elements move in a direction opposite to the derivational one, this algorithm's step (C) makes *EteNGraM* bidirectional, i.e. like BERT.

D. *Possible and impossible orders*

All combinations of 4 elements are 4! = 1*2*3*4 = 24. If *EteNGraM* makes correct predictions (properly reflects on the logic of how elements combine to form linear sequences in language), of the 24 possible combinations only orders containing the bigrams and trigrams listed in B and C above should exist in the languages of the world, while combinations of elements involving bad bigrams and trigrams (such that violate the assumptions in B and C) should not exist.

E. *Data clustering*

Different distributions of the data reveal different facts (recall the way Gauss clustered the numbers from 1 to 100 to find their sum). Thus, although Cinque structures the data as shown in (7), I will cluster the sequences around the opposition pre-positioning : post-positioning, (8) through (12) in section 4.

(7) Cinque's clustering of the data in the verbal domain (the same for the nominal domain)

a. √ Mood Tns Asp **V**
b. √ Mood Tns **V** Asp
c. √ Mood **V** Tns Asp
d. √ **V** Mood Tns Asp

e. (*) Tns Mood Asp **V**
f. (*) Tns Mood **V** Asp
g. * Tns **V** Mood Asp
h. * **V** Tns Mood Asp

i. (*) Asp Mood Tns **V**
l. (*) Asp Mood **V** Tns
m. √ Asp **V** Mood Tns
n. √ **V** Asp Mood Tns

o. * Mood Asp Tns **V**
p. √ Mood Asp **V** Tns
q. √ Mood **V** Asp Tns
r. * **V** Mood Asp Tns

s. * Tns Asp Mood **V**
t. √ Tns Asp **V** Mood
u. √ Tns **V** Asp Mood
v √ **V** Tns Asp Mood

w. (*) Asp Tns Mood **V**
x. * Asp Tns **V** Mood
y. √ Asp **V** Tns Mood
z. √ **V** Asp Tns Mood

There is no "j" because this letter is missing in the original text. The symbols after the letters mark existing "√", non-existing "*" and spurious "(*)" patterns.

4. Pre-training, fine-tuning and application of *EteNGraM*

In this section, I pretrain *EteNGraM* with the existing and non-existing TAM orders from Cinque (2014), then fine-tune the model and employ it for derivation of the Universal 20 and its exceptions.

As already mentioned, I will cluster the data around the opposition pre-positioning (prefixation) : post-positioning (suffixation). Thus, there are two series of four-element sequences, each starting with a default pattern of its own: the default pre-positioning pattern is #1, while the default post-positioning pattern is #13.

For ease of perception, the different types of bigrams and trigrams are colored as follows:

- Bigrams violating the principles in 3B and 3C are bad bigrams and are colored in red. A single bad bigram invalidates the whole sequence of four elements.
- Attested bad bigrams are in yellow, i.e. according to *EteNGraM* these bigrams should not exist. Attested bad bigrams are instrumental for the fine-tuning of the model.
- Hard to explain attested patterns are in cyan. Such patterns contain a trigram consisting of two bad bigrams. Again, such patterns will help us fine-tune the model.
- Stranded elements are in green; a stranded element does not have any impact on the well-formedness of a four-element sequence.
- Well-formed bigrams and trigrams (bigrams and trigrams derived according to the principles in 3B and 3C) are uncoloured.

4.1. Pre-training and fine-tuning of *EteNGraM*: Tns-Asp-Mood orders

(8) Pre-positioning, i.e. at least two elements precede the bolded base
Pre-positioning can be discussed as 1-2-3-**4**, with the base under "4", or as 4-3-2-**1**, with the base under "1". As demonstrated in this example, both orders lead to the same result, therefore I work only with the ascending order. The small letters following the numbering here and in (9) are those from Cinque's list in (7).

			base under "4" (bad bigrams in descending order)	base under "1" (bad bigrams in ascending order)
1.	a. √	Mood Tns Asp V	1-2-3-**4**	4-3-2-**1**
2.	b. √	Mood Tns V Asp	1-2-**4**-3	4-3-**1**-2
3.	p. √	Mood Asp V Tns	1-3-**4**-2	4-2-**1**-3
4.	t. √	Tns Asp V Mood	2-3-**4**-1	3-2-**1**-4
5.	e. (*)	Tns Mood Asp V	2-1-3-**4**	3-4-2-**1**
6.	f. (*)	Tns Mood V Asp	2-1-**4**-3	3-4-**1**-2
7.	i. (*)	Asp Mood Tns V	3-1-2-**4**	2-4-3-**1**

8.	l. (*)	Asp Mood V Tns	3-1-**4**-2		2-4-**1**-3
9.	w. (*)	Asp Tns Mood V	3-2-1-**4**	2-3-4-**1**	
10.	x. *	Asp Tns V Mood	3-2-**4**-1		2-3-**1**-4
11.	o. *	Mood Asp Tns V	1-3-2-**4**		4-2-3-**1**
12.	s. *	Tns Asp Mood V	2-3-1-**4**		3-2-4-**1**

As can be seen from (8), in pre-positioning *EteNGraM* successfully derives existing orders and bans non-existing and spurious patterns.

(9) Post-positioning, i.e. at least two elements follow the bolded base

13.	z. √	V Asp Tns Mood	**1**-2-3-4
14.	y. √	Asp V Tns Mood	2-**1**-3-4
15.	u. √	Tns V Asp Mood	3-**1**-2-4
16.	q. √	Mood V Asp Tns	4-**1**-2-3
17.	n. √	V Asp Mood Tns	**1**-2-4-3
18.	m. √	Asp V Mood Tns	2-**1**-4-3
19.	c. √	Mood V Tns Asp	4-**1**-3-2
20.	v √	V Tns Asp Mood	**1**-3-2-4
21.	d. √	V Mood Tns Asp	**1**-4-3-2
22.	h. *	V Tns Mood Asp	**1**-3-4-2
23.	g. *	Tns V Mood Asp	3-**1**-4-2
24.	r. *	V Mood Asp Tns	**1**-4-2-3

---

13 attested patterns altogether (out of 24)

In post-positioning (9), *EteNGraM* runs into trouble: sequences #17 through #21 (re-listed in (10) for convenience) should not exist but they do.

(10) Five false predictions, all with permutations of neighbor elements

13.	z. √	V Asp Tns Mood	**1**-2-3-4 (default)
17.	n. √	V Asp Mood Tns	**1**-2-4-3
18.	m. √	Asp V Mood Tns	2-**1**-4-3
19.	c. √	Mood V Tns Asp	4-**1**-3-2
20.	v √	V Tns Asp Mood	**1**-3-2-4
21.	d. √	V Mood Tns Asp	**1**-4-3-2 (2x permut. of -3- simultan., 4-3 and 3-2)

Intriguingly, all false predictions have one thing in common: contain permutations of legal bigrams of neighbor elements. #21 exhibits two simultaneous permutations of -3-.

Further scrutinizing (10), Tns (-3-), the middle element in the default post-positional sequence of non-base elements, Asp-Tns-Mood, #13 in (10), seems unfixed, in the sense that Tns precedes and follows both Asp (-2-) and Mood (-4-), therefore 2-3 / 3-2 and 3-4 / 4-3 bigrams in attested patterns, including the pattern with two permutations, #21. Recall that Tns behaves differently in pre-positioning and there are no permutations in (8). Cinque (2014: 18) makes a similar, but less powerful, observation: "As far as I have seen no reversals of Mood, Tense, and Aspect is attested with bona fide prefixes. Such orders thus seem not to represent genuine counterexamples to the order of Merge: (speech act) Mood > Tense > Aspect."

Now, it is obvious that *EteNGraM* needs fine-tuning and I will therefore add to the model the bigrams 3-2 and 4-3 as well-formed in post-positioning. (Clearly, BERT's fine-tuning is not the same as that of *EteNGraM* here but the architectures of the two models also significantly differ.)

4.2. Employing *EteNGraM* for derivation of Universal 20 and its exceptions

The letters following the numbers in (11) and (12) are those from Cinque (2005, ex. (6)) from where also the dataset was borrowed.

(11) Pre-positioning, i.e. at least two elements precede the base **4**

1.	a. √	Dem Num A N	1-2-3-**4**
2.	b. √	Dem Num N A	1-2-**4**-3
3.	n. √	Dem A N Num	1-3-**4**-2
4.	r. √	Num A N Dem	2-3-**4**-1
5.	e. *	Num Dem A N	2-1-3-**4**
6.	f. *	Num Dem N A	2-1-**4**-3
7.	i. *	A Dem Num N	3-1-2-**4**
8.	j. *	A Dem N Num	3-1-**4**-2
9.	u. *	A Num Dem N	3-2-1-**4**
10.	v. *	A Num N Dem	3-2-4-1
11.	m. *	Dem A Num N	1-3-2-**4**
12.	q. *	Num A Dem N	2-3-1-**4**

(12) Post-positioning, i.e. at least two elements follow the base **1**

13.	x. √	N A Num Dem	**1**-2-3-4	
14.	w. √	A N Num Dem	2-**1**-3-4	
15.	s. √	Num N A Dem	3-**1**-2-4	
16.	o. √	Dem N A Num	4-**1**-2-3	
17.	l. √	N A Dem Num	**1**-2-4-3	
18.	k. √	A N Dem Num	2-**1**-4-3	
19.	c. √	Dem N Num A	4-**1**-3-2	
20.	t. √	N Num A Dem	**1**-3-2-4	
21.	d. √	N Dem Num A	**1**-4-3-2	(2x permut. of 3 simultan., 4-3 and 3-2[2])
22.	g. *	Num N Dem A	3-**1**-4-2	
23.	h. *	N Num Dem A	**1**-3-4-2	
24.	p. (*)	N Dem A Num	**1**-4-2-3[3]	

[2] Cinque's (2014) writes the following in his "Fn 10: Greenberg (1963:87) states that N Dem Num A is "[a] less popular alternative" to N A Num Dem, citing Kikuyu as one example. Other languages that apparently display this order are Turkana, Rendille (Heine 1981), Noni (Hyman 1981:31), Nkore-Kiga (Lu 1998:162n59, 165), Abu' (Lynch 1998:171), Arbore (Hayward 1984:212), Bai and Moro (Dryer 2003:20, 43). It also appears as a possible alternative order in Romanian (Cornilescu 1992:212); but see Cinque 2004 for discussion."

[3] Cinque (2014) on this pattern: "Fn 27: The literature known to me reports only three languages with this order: Pitjantjatjara (Bowe 1990:111); Noni, which has (6d; 21 in our (12)) (N Dem Num A) as its primary order (Rijkhoff 2002:273); and Nkore-Kiga (Dryer 2003:43), which also has (6d, 21 in our (9)) as an alternative order (Lu 1998:162n59, 165). It is possible, thinking of its prevailing status as an alternative order, that this order is actually spurious (with A a reduced relative clause; see footnote 2) and that such subextractions should be ruled out entirely." The referred fn 2 is almost a page long and in its turn refers to fn 23 that is half a page.

---

13 attested patterns altogether (out of 24)

Like in the verbal domain, there are 13 attested patterns altogether. The fine-tuned *EteNGraM* now successfully predicts all patterns in pre-positioning (11) and in post-positioning (12). Actually, the patterns are exactly the same as those in the verbal domain, to the detail: In post-positioning in the verbal domain, the middle element in the default sequence of non-base elements, Asp-Tns-Mood, was unfixed; in the nominal domain, this element is Num, the default sequence is A-Num-Dem. Num may precede and follow A and Dem and there are 2-3 / 3-2 and 3-4 / 4-3 bigrams in attested orders in (12), including the trigram in #21 with two simultaneous permutations.

In sum, after fine-tuning (allowance of permutations of bigrams of neighbor elements in post-positioning), the bigrams and trigrams postulated by *EteNGraM* successfully derive all attested TAM and Universal 20 patterns and ban the unattested and spurious patterns. There are permutations of -3- with its neighbors postpositionally, thus 2-3 / 3-2 and 3-4 / 4-3; these permutations can happen simultaneously as well, giving the trigram 4-3-2. The bigram 4-2 (of non-neighbor elements) is the only bad bigram in post-position.

From a linguistic point of view, pre-positioning (prefixation) is regular, while post-positioning (suffixation) allows permutation of the middle non-base element in the default post-positional sequences, Tns in the verbal domain and Num in the nominal domain. The bigrams Mood-Asp and Dem-A are the only bad bigrams in post-positioning (one bad bigram in each domain). These two bigrams rule out all unattested combinations post-positionally in the verbal and nominal domain. In pre-positioning the bad bigrams tend to be at the beginning of a sequence (6 out of 8 cases). In post-positioning the bad bigram of a domain blocks 3 orders and is word-final in two of the cases. The verbal and the nominal domain are perfectly symmetrical.

5. Fine-tuning or what a linguist should learn from *EteNGraM*

Permutation of neighbor elements is so frequent in the languages of the world ((13) and (14)) that it was obvious that the model would need fine-tuning. I, however, decided to not "pre-program" permutation in order to demonstrate a process similar to fine-tuning in NLP.

(13) Yup'ik (Mithun 1999: 43)

*yug-**pag-cuar***	*yug-**cuar-pag***
person-big-little	person-little-big
'little giant'	'big midget'

There are many footnotes in Cinque (2005) and almost all are very long, I therefore refer the reader to the original text.

(14) English
(a) ***circular red*** *patch*[4]
red circular *patch*

(b) *I wrote* ***a letter to him****.*
I wrote ***him a letter****.*

It should be mentioned here that the fact that the sequences we have modeled do not exhibit permutation in prefixation does not mean that there does not exist a language with permutations in prefixation, see e.g. Rice (1989, 2000) for Athabaskan languages which are prefixing. But perhaps the most striking case of prefix permutation reported in the literature is Chinatang (Bickel et. al 2007) where we find two permutations simultaneously in prefixation (cf. #21 in both (9) and (12)).

(15) Chintang (Mulgãu) dialect (Bickel et. al 2007, p. 44; example clustering mine)
The abbreviations used in the glosses are as follows: NS nonsingular, A actor, P primary object, NEG negative, PST past.

(15.1) Two permutations of ***-kha-***
u-kha-ma-cop-yokt-e
3NS.ACTOR-1NS.P-NEG-see-NEG-PAST
ma-kha-u-cop-yokt-e
NEG-1NS.P-3NS.A-see-NEG-PST

(15.2) Two permutations of ***-ma-***
u-ma-kha-cop-yokt-e
3NS.A- NEG-1NS.P-see- NEG-PST
kha-ma-u-cop-yokt-e
1NS.P-NEG-3NS.A-see-NEG-PST

(15.3) Two permutations of ***-u-***
kha-u-ma-cop-yokt-e
1NS.P-3NS.A-NEG-see- NEG-PST
ma-u-kha-cop-yokt-e
NEG-3NS.A-1NS.P-see-NEG-PST

All meaning: 'They didn't see us.'

If the first examples in (15.1), (15.2) and (15.3) are thought to represent the base cases (I do not claim that they are the base cases, I have never learned Chintang), the second examples exhibit two permutations simultaneously of each of the three elements constituting these sequences of

[4] Larson (2021) proposes a revision of cartography and models the order of the English adjectives based on subjectivity ("salient, objective, factual properties of things vs. subjective properties"), the example in (14a) is from this study. I cannot agree with Larson, I think that linguistic analyses should rely on steady formal orientation points, such as these proposed in the present study, and that pushing towards syntactic features based on "extralinguistic relations" such as subjectivity is a move in the wrong direction for theoretical linguistics.

prefixes. The relevant question is now: Why does Chintang have two simultaneous permutations in prefixation, while the patterns we analyzed in section 4 do not. The answer seems to be: Tns-Asp-Mood and Universal 20 are prominent cross-linguistic patterns, while Chintang represents a language-specific extreme case, cf. also the categories involved in prefix permutation in Chintang.

It should also be mentioned that permutation seems quite common under stranding (see (9) and (12)), the only restriction is that the permuting elements must be neighbors, i.e. the difference in their numerical values cannot be >1. As explained in 3C, the stranded element does not violate the latter requirement because it is trapped and cannot move.

(16) Stranding + permutation (post-positioning, bigramic derivation)
2-**1**-3-4 → 2-**1**-4-3 (permutation of neighbor elements)
3-**1**-2-4 → *3-**1**-4-2 (bad bigram, permutation of non-neighbor elements)
4-**1**-2-3 → 4-**1**-3-2 (permutation of neighbor elements)

Recall that when I fine-tuned *EteNGraM*, I added the two bigrams, 3-2 and 4-3, to the list of legal bigrams.

In conclusion, although the fine-tuned *EteNGram* cannot account for Chintang (15) (and other prefixing languages), I do not think that it needs further fine-tuning at this stage. There are many other prominent sequences in linguistics (default orders of elements that define language linearly), one should first check the orders of elements in such sequences and only afterwards turn to language-specific orders like those in Chintang, although if analyzed in terms of two simultaneous permutations the prefix order in Chintang is no more exceptional. I suspect that the order of elements (words; NLP is not really interested in morphology) in prominent sequences is reducible to a limited number of linear patterns, similar to those I postulated for the TAM and Universal 20 orders. By contributing these patterns, linguists can significantly facilitate the development of the next-generation deep-net NLP architectures, in the sense that linguistic observations are expected to lead to reduction of the number of layers, and especially the number of self-attention heads in a DNN, cf. BERT's architecture in section 1. (On BERT's self-attention, see Kovaleva et al. 2019; Clark et al. 2019 discuss the type of linguistic information BERT seems to pay attention to.) Overall, collection of the combinatorial patterns of elements in prominent linear sequences seems a much more realistic research scenario for linguists than the one proposed by Linzen and Baroni (2021). The only problem I could see is how to avoid blurring the picture by language-particular patterns at the initial phases of this research.

6. Comparison of analyses

Let us compare now the complexity of *EteNGraM* uniform analysis of the TAM and Universal 20 orders (section 4) with Cinque's (2014) derivation of TAM patterns in (17), and with Cinque's (2005) derivation of Universal 20 in (18):

(17) *Derivation of TAM patterns* (cited from Cinque 2014)

a. Order of merge: [...[MoodP(speech act) Mood...[TensePTense...[AspPAspect [VPV]]]]]

b. Parameters of movement:

i) No movement, or

ii) VP movement without pied-piping, or

iii) VP movement plus pied-piping of the *whose pictures*-type, or

iv) VP movement plus pied-piping of the *pictures of who*-type

v) total vs. partial movement of the VP with or without pied-piping

vi) obligatory vs. optional application of movement.

vii) No movement of a phrase not containing the VP is possible (except for (focus) movements to the left of a second-position element).

(18) *Derivation of Universal 20 patterns* (cited from Cinque 2005)

a. Merge order: [. . . [$_{WP}$ Dem . . . [$_{XP}$ Num . . . [$_{YP}$ A [$_{NP}$ N]]]]]

b. Parameters of movement:

i. No movement (unmarked), or

ii. Movement of NP plus pied-piping of the *whose picture* type (unmarked) or

iii. Movement of NP without pied-piping (marked), or

iv. Movement of NP plus pied-piping of the *picture of who* type (more marked still).

v. Total (unmarked) versus partial (marked) movement of NP with or without pied-piping (in other words, NP raises all the way up, as in (d,l,p,t,x), or just partially, as in (b,c,k,n,o,r,s,w), around its modifiers).

vi. Neither head movement nor movement of a phrase not containing the (overt) NP is possible (except perhaps for focus-related movements of phrases to a DP-initial position).

Note the different number of derivational steps in (17) and (18): seven (vii) steps in the verbal domain and only six (vi) in the nominal domain. Compare (17) and (18) with *EteNGraM'*s uniform derivation of the TAM and Universal 20 orders in section 4.

The long citations from Cinque (2014) and Cinque (2005) below are only for orientation. The many footnotes referring to these citations in the original texts are deleted here. Readers interested in Cinque's exact explanations of the derivations of the TAM and Universal 20 orders, should check Cinque (2014) and Cinque (2005), respectively.

Cinque's explanation of the attested TAM orders

As mentioned many times, my presentation of the data is based on pre- and post-positioning, therefore Cinque's alphabetic list in (7), is not sequential in (8) and (9). Nevertheless, in (8) and (9) Cinque's **a** corresponds to a, **b** to b, etc., which also holds for the quotation below:

> **a** is derived if nothing moves;
>
> **b** is derived if VP raises to a Spec between Tense and Aspect, with no further movement involved (Mood Tns VP_k Asp t_k);
>
> **c** is derived if the VP moves further to a Spec between Mood and Tense (Mood VP_k Tns (t_k) Asp t_k);
>
> **d** is derived if the VP moves further to a Spec higher than Mood (VP_k Mood (t_k) Tns (t_k) Asp t_k);
>
> **m** is derived if VP moves to a Spec higher than Mood pied piping the projection dominating Asp ([Asp VP]$_i$ Mood (t_i) Tns t_i);
>
> **n** is derived if VP raises to a Spec between Tense and Aspect, and then raises to a Spec higher than Mood pied piping the projection dominating it and Aspect ([VP_k Asp t_k]$_i$ Mood (t_i) Tns t_i);
>
> **p** is derived if VP moves to a Spec between Mood and Tense pied piping the projection dominating Asp (Mood [Asp VP]$_i$ Tns t_i);
>
> **q** is derived if VP raises to a Spec between Tense and Aspect, and then raises to a Spec between Mood and Tense pied piping the projection dominating it and Asp (Mood [VP_k Asp t_k]$_i$ Tns t_i);

t is derived if VP moves to a Spec higher than Mood pied piping the projection dominating Tense, Aspect and VP ([Tns Asp VP]$_i$ Mood t_i);

u is derived if VP raises to a Spec between Tense and Aspect, and then raises to a Spec higher than Mood pied piping the projection dominating Tense, VP and Aspect ([Tns VP_k Asp t_k]$_i$ Mood t_i);

v is derived if VP raises to a Spec between Mood and Tense, and then raises to a Spec higher than Mood pied piping the projection dominating it, Tense and Aspect ([VP_k Tns (t_k) Asp t_k]$_i$ Mood t_i);

y is derived if VP moves to a Spec between Mood and Tense pied piping the projection dominating Aspect, and then raises to a Spec higher than Mood pied piping the projection dominating Aspect, VP and Tense ([[Asp VP]$_i$ Tns t_i]$_j$ Mood t_j]);

z is derived if VP moves to a Spec between Tense and Aspect, then moves to to a Spec between Mood and Tense pied piping the projection dominating VP and Aspect, and then raises to a Spec higher than Mood pied piping the projection dominating it, Aspect and Tense ([[[[VP_i Asp t_i]$_j$ Tns t_j]l Mood t_l]).

Cinque (2014) does not provide any explanation of the unattested TAM orders.

Cinque's explanation of the Universal 20 patterns, attested and unattested

In our (11) and (12), Cinque's (2005) (a) corresponds to a, (b) to b, etc. In the quotation below, (18) is referred to many times, this is (18) *Derivation of Universal 20 patterns* (cited from Cinque 2005) in this section; (18) was inserted in the original text for reader's convenience; references to footnotes are deleted:

(**a**) (Dem Num A N) is derived if nothing moves (18bi). (No marked option: very many languages.)

(**b**) (Dem Num N A) is derived from Dem Num A N if NP raises one notch, around A, either with (vacuous) pied-piping of the *whose picture* type (18bii) (unmarked) or without pied-piping (18biii) (marked). (Despite the markedness of partial movement, it includes the unmarked case of pied-piping: many languages.)

(**c**) (Dem N Num A) is derived if NP moves two notches, around A and Num (i.e., par-tially—marked option) without pied-piping ((18biii)—marked option). (Two marked options: very few languages.)

(**d**) (N Dem Num A) is derived if NP moves three notches, around A, Num, and Dem (i.e., all the way up) without pied-piping ((18biii): marked). (One marked option: few languages.)

(**e**) (Num Dem A N) cannot be derived through (18). NP has not moved, and the modifiers to its left are in the wrong Merge order (cf. (18a)).

(**f**) (Num Dem N A) cannot be derived through (18). Raising of NP without pied-piping implies a wrong Merge order of the modifiers (Num Dem A N) (see (18a)). Raising of NP with pied-piping of the *picture of who* type either of [Dem N] or of [Num Dem N] also implies a wrong Merge order (either Num A [Dem N] or A [Num Dem N]).

(**g**) (Num N Dem A) cannot be derived through (18). Raising of NP without pied-piping implies that the Merge order is Num Dem A N, which is a wrong order. Raising of NP with pied-piping of the *whose picture* type again implies a wrong Merge order of the modifiers (Num A Dem N), with N first raising around Dem and [N Dem] then raising around A. Raising of NP with pied-piping of the *picture of who* type (raising of [Num N] two notches) also implies a wrong Merge order of the modifiers (Dem A Num N).

(**h**) (N Num Dem A) cannot be derived through (18). Raising of NP without pied-piping implies a wrong Merge order (Num Dem A N). Raising of NP with successive pied-pipings of the *whose picture* type also implies a wrong Merge order (A Dem Num N). Raising of NP without pied-piping around Dem and Num, followed by raising with pied-piping around A, would derive(6h), but, again, from a wrong Merge order (A Num Dem N). (Similarly if NP were to move around Num and pied-pipe it to the left of A and then move on without further pied-pipings. TheMerge order in this case would be Dem A Num N—again, the wrong order.)

(**i**) (A Dem Num N) cannot be derived through (18). NP has not moved, and the modifiers to its left are in the wrong Merge order (see (18a)).

(**j**) (A Dem N Num) cannot be derived through (18). NP has moved one notch, but the two modifiers to its left are in the wrong Merge order (see (18a)). (j) could also arise via raising of NP with pied-piping of the *picture of who* type of either Dem N or A Dem N around Num, but both derivations presuppose a wrong Merge order (A Num Dem N and Num A Dem N, respectively).

(**k**) (A N Dem Num) has a well-formed, though marked, derivation with raising of NP plus pied-piping of the *picture of who* type of the lowest modifier (A), followed by raising of [A N]without pied-piping around both Num and Dem. (Two marked options: very few languages.)

(**l**) (N A Dem Num) has a derivation in which NP raises past A, followed by pied-piping of the *whose picture* type past Num, followed by raising of [N A] without pied-piping (marked) past Dem. (One marked option: few languages.)

(**m**) (Dem A Num N) cannot be derived through (18). NP has not moved, and the modifiers to its left are in the wrong Merge order (see (18a)).

(**n**) (Dem A N Num) has a derivation with partial (marked) raising of NP plus pied-piping of the *picture of who* type of [A N] (marked) around Num. (Two marked options: very few languages.)

(**o**) (Dem N A Num) has a derivation from (18a) involving partial (marked) raising of NP plus pied-piping of the *whose picture* type, vacuously, and non vacuously (of [N A]) around Num. (One marked option: many languages.)

(**p**) (N Dem A Num), if genuine (see our footnote 2), may be especially marked, as its derivation from (10a) would seem to involve raising of NP with successive pied-pipings of the *whose picture* type around A and Num (alternatively, a single raising of the *picture of who* type of [A N] around Num) and then extraction of the sole NP around Dem.

(**q**) (Num A Dem N) cannot be derived through (18). NP has not moved, and the modifiers to its left are in the wrong Merge order (see (18a)).

(**r**) (Num A N Dem) has a derivation with partial (marked) raising of NP plus pied-piping of the *picture of who* type of A and Num ([Num A N]) (marked) around Dem. (Two marked options: very few languages.)

(**s**) (Num N A Dem) has a derivation with partial (marked) raising of NP around A, followed by raising plus pied-piping of the *picture of who* type of [Num N A] (marked) around Dem. (Two marked options: few languages.)

(**t**) (N Num A Dem) has a derivation with raising of NP without pied-piping around A and Num (marked), followed by raising plus pied-piping of the *whose picture* type of [N Num A] around Dem. (One marked option: few languages.)

(**u**) (A Num Dem N) cannot be derived through (18). NP has not moved, and the modifiers to its left are in the wrong Merge order (see (18a)).

(**v**) (A Num N Dem) cannot be derived through (18). Raising of NP without pied-piping implies a wrong Merge order of the modifiers (A Num Dem N) (see (18a)). Raising of NP with pied-piping of the *picture of who* type either of [Num N] or of [A Num N] also implies a wrong Merge order (either A Dem [Num N] or Dem [A Num N]).

(**w**) (A N Num Dem) has a derivation from (18a) with raising of NP plus pied-piping of the *picture of who* type of A around Num (marked), followed by raising of [A N Num] around Dem. (One marked option: few languages.)

(**x**) (N A Num Dem) has a derivation from (18a) involving raising of NP with successivepied-pipings of the *whose picture* type all the way up. (No marked option: very many languages.)

The fact that all N-final orders that do not respect the order Dem Num A ((e), Num Dem A N; (i), A Dem Num N; (m), Dem A Num N; (u), A Num Dem N) are very clearly unattested can indeed be taken to indicate that it is the raising of NP (or of an XP containing it) that is responsible for word order variation within the DP (perhaps, more generally, that it is the raising of the lexical part of a phrase that is responsible for word order variation within its ‘‘extended projection’’).

Since my goal has not been to (re-)formulate a theory but to check whether it is possible to do NLP without grammar and to solve a linguistic problem in the simplest possible way, I will not provide a critical analysis of Cinque’s derivations. Moreover, there are many alternative explanations of Greenberg’s Universal 20 (and Cinque’s analysis):

1) Dryer (2006) proposes an analysis in terms of general principles of symmetry and harmony;
2) Abels & Neeleman's (2009) analysis is based on Kayne's (1994) Linear Correspondence Axiom;
3) Cysouw (2010) speaks of observable preferences for head positions: "noun and adjective tend to occur together, nouns and demonstratives prefer to occur at the phrase boundary, and noun-adjective order is slightly more frequent than adjective-noun order" (This analysis is positional and focuses on the linear order of elements, i.e. it is in accord with *EteNGraM* but its predictions are less powerful than *EteNGraM*'s and, consequently, Cysouw pleads for probabilistic modeling of linguistic diversity, while *EteNGram* classifies in terms of possible and impossible patterns);
4) Steddy & Samek-Lodovici (2011) propose a solution in terms of four Optimality Theory constraints requiring leftward alignment of the items involved;
5) Steedman (2011) speaks of primitive operations of combinatory categorial grammars;
6) Culbertson & Adger (2014) see Universal 20 as possibly reflecting a deep property of the human cognitive system: "noun phrase structure is likely represented not primarily in terms of linear order, but rather in terms of hierarchical relations encoding semantic scope."
7) Merlo (2015) and Merlo & Ouwayda (2018) model Universal 20 in terms of different types of syntactic movement.

Evaluating all these proposals is outside the scope of this study. Nevertheless, it is safe to say that all linguistics solutions to Universal 20 are syntactic in nature, in one way or another; they are thus more complex than the bigram- and trigram-based *EteNGraM*.

It should also be mentioned here that solutions involving syntactic trees are hard to understand by computer scientists which could be one of the reasons why syntactic trees are disperefered in NLP research. I will only outline the issue, a detailed discussion of trees and templates in CS and in linguistics in Manova (ms.).

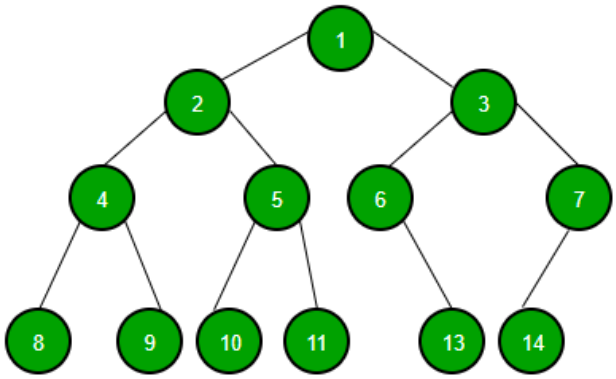
A computer works with templatically organized sequences of zeros and ones, i.e. everything (including all types of data structures, hierarchical and nonhierarchical alike) ends up as a linear sequence of zeros and ones. Templates are sequences of e.g. 16, 32 or 64 positions (bits) and every position has a specific predefined value (on the positional nature of language, see Manova et al. 2020). For example, a 16-bit template has predefined values of positions as shown in table 1.

As can be seen from table 1, if "1" occupies one of the 16-bit template positions, the final value of the position remains unchanged (= 1*position value), if "0" occupies a position, the value of the position becomes zero because multiplication by zero gives zero. As for the representations of the decimal numbers, e.g. $10 = 2^3 + 2^1 = 8 + 2$.

Table 1: A 16-bit template, i.e. this is how a 16-bit machine computes all types of tasks

Bit #	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Predefined value	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
Representation of decimal 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Representation of decimal 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Representation of decimal 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
...																
Representation of decimal 10	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
...																

Diagram 1: Binary tree (CS)



Source: https://www.geeksforgeeks.org/binary-tree-data-structure/

As is well-known from generative grammar, trees are a derivational device in linguistics. By contrast, in CS, trees serve for storage and organization of information, and programs are created on how to read these trees, i.e. a CS tree is compatible with different readings (while a syntactic tree always has a single reading). There are also different predefined types of trees in CS: rooted and unrooted trees, binary and non-binary trees, balanced and unbalanced trees, and so on. For example, in a rooted balanced binary tree, the height of the left and right subtree of any node differ by not more than one, i.e. Diagram 1 illustrates a rooted balanced binary tree. Additionally, all nodes of a rooted tree are numbered in a strict order starting from the root: in Diagram 1 the root node is under "1". In a rooted tree, the root serves as an orientation point, in the sense that the creation of the tree and its search start from the root. Since the drawing of a

syntactic tree does not start from the root but from the most distant leaf (called branch in syntax), trees in syntax appear unrooted but with roots. The rooted-unrooted issue with syntactic trees is best visible in Distributed Morphology (Halle & Marantz 1993; Harley & Noyer 1999; Bobaljik 2017, among many others) where the root of the word is not in the root of the tree. Thus from a CS point of view, with the version of trees used in syntax (i.e. trees that are unrestricted in terms of height and weight and that permit different types of movement of elements) everything can be derived.

Interim conclusion, one can solve the same linguistic problem such as e.g. derivation of Universal 20 and its exceptions in different ways, the different solutions, although leading to the same result (Cinque's list of attested and unattested patterns), differ in complexity. That is, complexity in linguistics is like complexity in CS -- a property of the solution (analysis). The derivation of the cross-linguistic TAM orders and Universal 20 and its exceptions do not require hierarchical structures such as syntactic trees. These two prominent linguistic sequences are linear structures and as such they can be derived only with bigrams and trigrams.

7. Additional data for Universal 20

Recently additional data for Universal 20 have been published (Merlo & Ouwayda 2018). In this section I challenge the *EteNGraM* model with these data. The goal is, among other things, to check whether *EteNGraM*'s high efficiency is not by chance.

Table 2 is adopted from Merlo & Ouwayda (2018), with a single modification: the first column is added by me and contains the numbering of the Universal 20 patterns from (11) and (12). For convenience, (11) and (12) are repeated at the end of this section. The rest of the columns are named as in the original version of the table: "Dryer's Languages" and "Dryer's Genera" give the counts reported in Dryer (2006) by language and by genera respectively; the next two columns "Cinque's 05 Languages" and "Cinque's 13 Languages" countain Cinque's counts from 2005, i.e. what we analyzed and confirmed in section 4, as well as Cinque's (2013) database, a private resource shared with the authors of table 2, Merlo and Ouwayda. The patterns reported as non-existing or spurious in Cinque (2005) and confirmed as non-existing by *EteNGraM* but existing according to Dryer (2006) are shaded in Table 2 where all bases are also colored. The goal of this colorful presentation is to show that, like in Cinque (2005), Merlo & Ouwayda's (2018) major concern seemed not to be the appropriate clustering of the data, such that favors a solution to the problem, but avoidance of omission of a pattern, although one could also interpret their approach as 'monitoring the movement of N'. Roughly, they change the placement of the N-column one position forward each time, see Table 2; and explain the different patterns as types of movement with different costs of the syntactic operations involved.

Table 2: = "Table 1: Attested word orders of Universal 20 and their estimated frequencies" from Merlo & Ouwayda (2018); column 1 and colors added by SM

# in (11) and (12)					Dryer's Languages	Dryer's Genera	Cinque's 05 Languages	Cinque's 13 Languages
1	DEM	NUM	ADJ	N	74	44	V. many	300
11	DEM	ADJ	NUM	N	3	2	0	0
5	NUM	DEM	ADJ	N	0	0	0	0
12	NUM	ADJ	DEM	N	0	0	0	0
7	ADJ	DEM	NUM	N	0	0	0	0
9	ADJ	NUM	DEM	N	0	0	0	0
2	DEM	NUM	N	ADJ	22	17	Many	114
3	DEM	ADJ	N	NUM	11	6	V. few (7)	35
6	NUM	DEM	N	ADJ	0	0	0	0
4	NUM	ADJ	N	DEM	4	3	V. few (8)	40
8	ADJ	DEM	N	NUM	0	0	0	0
10	ADJ	NUM	N	DEM	0	0	0	0
16	DEM	N	ADJ	NUM	28	22	Many	125
19	DEM	N	NUM	ADJ	3	3	V. few (4)	37
22	NUM	N	DEM	ADJ	5	3	0	0
15	NUM	N	ADJ	DEM	38	21	Few (2)	180
18	ADJ	N	DEM	NUM	4	2	V. few (3)	14
14	ADJ	N	NUM	DEM	2	1	V. few	15
21	N	DEM	NUM	ADJ	4	3	Few (8)	48
24	N	DEM	ADJ	NUM	6	4	V. few (3)	24
23	N	NUM	DEM	ADJ	1	1	0	0
20	N	NUM	ADJ	DEM	9	7	Few (7)	35
17	N	ADJ	DEM	NUM	19	11	Few (8)	69
13	N	ADJ	NUM	DEM	108	57	V. many (27)	411

Four of the 24 patterns in Table 2 seem problematic for *EteNGraM*: these patterns should not exist but they seem to do. Note, however, that Cinque's (2013) database confirms only one of the four problematic patterns, #24 in (11): N Dem A Num 1-4-2-3. Cinque (2013) reports 24 languages of this type versus only 6 in Dryer (2006), and 3 in Cinque (2005), but see footnote 3 where Cinque explains why the three languages should be considered spurious instances of the pattern. The other three problematic patterns occur in a very limited number of languages: N

Num Dem A (#23 in (12)) in only 1 language; Dem A Num N (#11 in (11)) in only 3 languages; and Num N Dem A (#22 in (12)) in only 5 languages. All counts according to Dryer (2006), as already mentioned, for Cinque (2005, 2013) these patterns do not exist. Since the languages violating *EteNGraM's* derivational rules are not well-studied, it is hard to say what is behind these violations: Do they really exist or are they simply due to insufficient knowledge of the target language by the person providing the description.

Thus, only pattern #24: N Dem A Num seems to pose a real problem to *EteNGraM*. However, with respect to this pattern there is a significant discrepancy in the counts in Cinque (2005), Dryer (2006) and Cinque (2013): 3 versus 6 versus 24 languages. This situation seems to indicate definitional and/or methodological inconsistencies.

(11 repeated) Pre-positioning, i.e. at least two elements precede the base **4**

1.	a. √	Dem Num A N	1-2-3-**4**
2.	b. √	Dem Num N A	1-2-**4**-3
3.	n. √	Dem A N Num	1-3-**4**-2
4.	r. √	Num A N Dem	2-3-**4**-1
5.	e. *	Num Dem A N	2-1-3-**4**
6.	f. *	Num Dem N A	2-1-**4**-3
7.	i. *	A Dem Num N	3-1-2-**4**
8.	j. *	A Dem N Num	3-1-**4**-2
9.	u. *	A Num Dem N	3-2-1-**4**
10.	v. *	A Num N Dem	3-2-4-1
11.	m. *	Dem A Num N	1-3-2-**4**
12.	q. *	Num A Dem N	2-3-1-**4**

(12 repeated) Post-positioning, i.e. at least two elements follow the base **1**

13.	x. √	N A Num Dem	**1**-2-3-4	
14.	w. √	A N Num Dem	2-**1**-3-4	
15.	s. √	Num N A Dem	3-**1**-2-4	
16.	o. √	Dem N A Num	4-**1**-2-3	
17.	l. √	N A Dem Num	**1**-2-4-3	
18.	k. √	A N Dem Num	2-**1**-4-3	
19.	c. √	Dem N Num A	4-**1**-3-2	
20.	t. √	N Num A Dem	**1**-3-2-4	
21.	d. √	N Dem Num A	**1**-4-3-2	(2x permut. of 3 simultan., 4-3 and 3-2)
22.	g. *	Num N Dem A	3-**1**-4-2	
23.	h. *	N Num Dem A	**1**-3-4-2	
24.	p. (*)	N Dem A Num	**1**-4-2-3	

8. The full power of *EteNGraM*

In this section, I will demonstrate that *EteNGraM* allows for fine-grained classification of the attested patterns of Universal 20 and that, consequently, it can be used for making predictions about preferred and dispreferred patterns across languages.

Table 3: = Table 2 but with the data clustering of (11) and (12), both repeated at the end of the previous section

The colours used in the table are those from (11) and (12): stranded elements do not violate anything; bad bigrams invalidate whole patterns, and permutations of neighbor elements in post-positioning do not violate anything

# in (11) and (12) and a comment					Dryer's Languages	Dryer's Genera	Cinque's 05 Languages	Cinque's 13 Languages
Pre-positioning								
1 (default)	**DEM**	**NUM**	**ADJ**	**N**	**74**	**44**	**V. many**	**300**
2 (stranding)	DEM	NUM	N	ADJ	22	17	Many	114
3 (stranding)	DEM	ADJ	N	NUM	11	6	V. few (7)	35
4 (stranding)	NUM	ADJ	N	DEM	4	3	V. few (8)	40
5 (bad bigram)	NUM	DEM	ADJ	N	0	0	0	0
6 (bad bigram)	NUM	DEM	N	ADJ	0	0	0	0
7 (bad bigram)	ADJ	DEM	NUM	N	0	0	0	0
8 (bad bigram)	ADJ	DEM	N	NUM	0	0	0	0
9 (bad bigram)	ADJ	NUM	DEM	N	0	0	0	0
10 (bad bigram)	ADJ	NUM	N	DEM	0	0	0	0
11 (bad bigram)	DEM	ADJ	NUM	N	3	2	0	0
12 (bad bigram)	NUM	ADJ	DEM	N	0	0	0	0
Post-positioning								
13 (default)	**N**	**ADJ**	**NUM**	**DEM**	**108**	**57**	**V. many (27)**	**411**
14 (stranding)	ADJ	N	NUM	DEM	2	1	V. few	15
15 (stranding)	NUM	N	ADJ	DEM	38	21	Few (2)	180
16 (stranding)	DEM	N	ADJ	NUM	28	22	Many	125
17 (permutation)	N	ADJ	DEM	NUM	19	11	Few (8)	69
18 (strand+permut)	ADJ	N	DEM	NUM	4	2	V. few (3)	14
19 (strand+permut)	DEM	N	NUM	ADJ	3	3	V. few (4)	37
20 (permutation)	N	NUM	ADJ	DEM	9	7	Few (7)	35
21 (2 permutations)	N	DEM	NUM	ADJ	4	3	Few (8)	48
22 (bad bigram)	NUM	N	DEM	ADJ	5	3	0	0
23 (bad bigram)	N	NUM	DEM	ADJ	1	1	0	0
24 (bad bigram)	N	DEM	ADJ	NUM	6	4	V. few (3)	24

Scrutinizing table 3:

1) Pre-positioning

Stranding of the closest to the default base element, ADJ (#2), is more preferred than stranding of the most distant elemment, DEM (#4), 114 versus 40 languages in Cinque (2013).

2) Post-positioning

Stranding without permutation (#14, #15, and #16) is more preferred than stranding with permutation (#18 and #19). Now recall what we established with respect to fixedness of the non-base elements in post-positioning (section 4): NUM is unfixed, ADJ and DEM are fixed, i.e. all permutations are due to NUM. Stranding of an unfixed element (#15) is more preferred than stranding of a fixed element (#14 and #16). Stranding of the most distant from the base (fixed) element (# 16) is more prefered than stranding of the closest to the base (fixed) element (#14).

Permutation patterns are extracted from table 3 and listed as a separate table 4 for ease of perception.

Table 4: Permutation patterns in post-positioning (based on table 3)

13 (default)	**N**	**ADJ**	**NUM**	**DEM**	**108**	**57**	**V. many (27)**	**411**
17 (permutation)	N	ADJ	DEM	NUM	19	11	Few (8)	69
18 (strand+permut)	ADJ	N	DEM	NUM	4	2	V. few (3)	14
19 (strand+permut)	DEM	N	NUM	ADJ	3	3	V. few (4)	37
20 (permutation)	N	NUM	ADJ	DEM	9	7	Few (7)	35
21 (2 permutations)	N	DEM	NUM	ADJ	4	3	Few (8)	48

As can be seen from table 4, permutation of elements distant from the default base (#17) is more preferred than permutation of elements closer to the default basse (#20), 69 versus 35 languages in Cinque (2013). Permutation without stranding (#17) is more preferred than the same permutation under stranding (#18), 69 versus 14 languages. Under stranding (#18 and #19), the combinatorial logic appears to be the opposite to without stranding: permutation of elements closer to the default base (#19) seems more preferred than permutation of more distant elements (#18), 37 versus 14 languages, respectively. The relatively high frequency of pattern #21, with two simultaneous permutations (according to Cinque 2013, 48 languages tolerate two simultaneous permutations), indicates that the major ordering factor is distance / closeness to the default base and not permutation. This observation finds support by what has been established for pre- and post-positioning without permutation. In prepositioning, stranding of the closest to the default base element (#2) is more preferred than stranding of the most distant elemment (#4), 114 versus 40 languages; and the reverse rule in post-positioning, stranding of the most distant from the default base element (#16) is more prefered than stranding of the closest to the base element (#14), 125 versus 15 languages.

8. Conclusions

To check whether NLP without grammar is possible, I created a very simple NLP model, *EteNGraM*, that imitates current NLP research by handling semantic relations without semantics. I pre-trained the model with the TAM orders in the verbal domain, fine-tuned it and employed it

for derivation of Universal 20 and its exceptions in the nominal domain. Although *EteNGraM* is ridiculously simple (it derives language structure only with linear sequences such as bigrams and trigrams), it makes exactly the same predictions with respect to existing and non-existing TAM and Universal 20 orders as those in Cinque (2005, 2014). After a comparison of *EteNGram*'s uniform analysis of the TAM and Universal 20 orders with Cinque's syntax-based analyses of these orders and with other proposals from the literature for derivation of Universal 20, I briefly addressed the nature of trees in CS and in syntax and demonstrated that syntactic trees are problematic as a derivational device. *EteNGraM* also revealed novel facts about the organization of the verbal and nominal domain: pre-positioning is regular in both domains, while post-positioning tolerates permutation of neighbor elements, also in both domains; there is a perfect symmetry of existing and non-existing patterns in both domains and in the ways these patterns are derived; the same (numerically, i.e. 4-2) bad bigram is responsible for all non-existing patterns in post-positioning in both domains; in post-positioning in both domains, the middle non-base element (Tns and Num, respectively) is unfixed, which thus explains why there are no permutations in pre-positioning. To demonstrate that the surprisingly successful performance of *EteNGram* is not by chance, I challenged the model with additional data for Universal 20. *EteNGraM* not only derived these data but also made fine-grained predictions about preferred and dispreferred patterns across languages based on features such as +/-stranding, stranding of +/-fixed element, stranding with and without permutation, closeness to / distance from the base, etc. *EteNGraM* also revealed the necessity for bidirectional analyses of language data, which is the case in current NLP research. Therefore, establishing the combinatorial patterns of elements (words) in prominent linear sequences in terms of n-grams seems a promising scenario for a fruitful dialogue with the NLP community. Clearly, *EteNGraM* can be applied for derivation of sequences longer than those discussed in this paper; this is planned for future research.

References

Abels, K. & A. Neeleman. 2009. "Universal 20 without the LCA." In *Merging features: Computation, interpretation, and acquisition*, edited by J. M. Brucart, A. Gavarró & J. Solà, 60–79. Oxford: Oxford University Press. DOI: https://doi.org/10.1093/acprof:oso/9780199553266.003.0004

Baayen, R. H., I. Milin & D. Filipovic Đurđevic, P. Hendrix & M. Marelli. 2011. "An amorphous model for morphological processing in visual comprehension based on naive discriminative learning." *Psychological Review* 118. 438–481.

Baker, M. 1985. "The Mirror Principle and morphosyntactic explanation". *Linguistic Inquiry* 16: 373-415.

Baroni, M. 2021. "On the proper role of linguistically-oriented deep net analysis in linguistic theorizing." https://ling.auf.net/lingbuzz/006031

Bobaljik, J. 2017. "Distributed Morphology." *Oxford research encyclopedia of linguistics*. Retrieved 16 Jun. 2021, from https://oxfordre.com/linguistics/view/10.1093/acrefore/9780199384655.001.0001/acrefore-9780199384655-e-131.

Bybee, J. 1985. *Morphology. A study of the relation between meaning and form*. Amsterdam, Benjamins

Cinque, G. 2005. “Deriving Greenberg's Universal 20 and Its Exceptions” *Linguistic Inquiry* 36: 315-332.

Cinque, G. 2014. “Again on Tense, Aspect, Mood Morpheme Order and the “Mirror Principle”.” In *Functional Structure from Top to Toe: The Cartography of Syntactic Structures*, *Volume 9,* ed. by P. Svenonius. Oxford University Press.

Clark, K., U. Khandelwal, O. Levy, C. D. Manning. 2019. "What Does BERT Look at? An Analysis of BERT's Attention". *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Stroudsburg, PA, USA: Association for Computational Linguistics: 276–286. doi:10.18653/v1/w19-4828.

Culbertson, J. & D. Adger. 2014. “Language learners privilege structured meaning over surface frequency.” *Publications of the National Academy of Science* 111(16). 5842–5847. DOI: https://doi.org/10.1073/pnas.1320525111

Cysouw, M. 2010. “Dealing with diversity: Towards an explanation of NP word order frequencies.” *Linguistic Typology* 14(2). 253–287. DOI: https://doi.org/10.1515/lity.2010.010

Greenberg, J. 1963. “Some universals of grammar with particular reference to the order of meaningful elements.” In *Universals of language*, ed. by J. Greenberg, 73–113. Cambridge, Mass.: MITPress.

Greenberg, J. 1989. “The internal and external syntax of numerical expressions: Explaining language-specific rules.” In *Universals of language*, ed. by M. Kefer and J. van der Auwera, 105–118. (Belgian Journal of Linguistics 4.) Brussels: Editions de l’Universite ́de Bruxelles.

Devlin, J., M.-W. Chang, K. Lee & K. Toutanova. 2018. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” https://arxiv.org/abs/1810.04805

Dryer, M. S. 1992. “The Greenbergian word order correlations.” *Language* 68. 81–138. DOI: https://doi.org/10.2307/416370

Dryer, M. S. 2006. “The order demonstrative, numeral, adjective and noun: An alternative to Cinque.” http://exadmin.matita.net/uploads/pagine/1898313034_cinqueH09.pdf.

Halle, M. & A. Marantz. 1993. “Distributed morphology and the pieces of inflection.” In , *The view from building 20*, edited by K. Hale & S. J. Keyser, 111–176. Cambridge, MA: MIT Press.

Harley, H. & R. Noyer. 1999. “Distributed Morphology.” *Glot International*, Volume 4, Issue 4.

Hawkins, J. A. 1983. *Word order universals. Quantitative Analyses of Linguistic Structure.* New York; London: Academic Press.

Jurafsky, D. & J. H. Martin. 2020. *Speech and Language Processing*. 3rd Edition. Draft of December 30, 2020, https://web.stanford.edu/~jurafsky/slp3/

Kayne, R. 1994. *The antisymmetry of syntax*. Cambridge, MA: MIT Press.

Kovaleva, O, A. Romanov, A. Rogers, A. Rumshisky. 2019. "Revealing the Dark Secrets of BERT". *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 4364–4373.

Larson, R. 2021. “Rethinking Cartography.” *Language* 97(2): 245-268.

Linzen, T. & M. Baroni. 2021. “Syntactic Structure from Deep Learning.” *Annual Review of Linguistics 2021* 7(1): 195-212, https://ling.auf.net/lingbuzz/005161

Manova, S. 2019. “Writing texts with n-grams: The Fibonacci sequence in linguistics.” *Lectures in Cognitive Science*, University of Vienna, October 15, 2019.
Manova, S. 2020. “Natural language processing with n-grams: A linguistic solution to the zero-frequency problem.” *Lectures in Cognitive Science*, University of Vienna, November 3, 2020.
Manova, S. (to appear). “Ordering restrictions between affixes.” In *The Wiley Blackwell Companion to Morphology*, edited by P. Ackema, S. Bendjaballah, E. Bonet & A. Fábregas. Blackwell.
Manova, S. (ms). “Why are templates more powerful than syntactic trees?” Ms. University of Vienna.
Manova, S., H. Hammarström, I. Kastner & Y. Nie. 2020. “What is in a morpheme? Theoretical, experimental and computational approaches to the relation of meaning and form in morphology.” *Word Structure* 13(1): 1-21.
Manova, S. & G. Knell. 2021. “Two-suffix combinations in native and non-native English; Novel evidence for morphomic structures.” In *All Things Morphology*: *Its independence and its interfaces*, edited by S. Moradi, M. Haag, J. Rees-Miller & A. Petrovic. Amsterdam: Benjamins, 305-323. https://ling.auf.net/lingbuzz/005725
Mansfield, J., S. Stoll & B. Bickel. 2020. “Category clustering: A probabilistic bias in the morphology of verbal agreement marking.” *Language* 96 (2): 255-293, https://muse.jhu.edu/article/757629/pdf
McCoy, R. T., J. Min & T. Linzen. 2020. “BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance.” *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 217–227. Association for Computational Linguistics.
Merlo, P. 2015. “Predicting word order universals.” *Journal of Language Modelling* 3(2). 317–344. DOI: https://doi.org/10.15398/jlm.v3i2.112
Merlo, P., & Ouwayda, S. 2018. “Movement and structure effects on Universal 20 word order frequencies: A quantitative study.” *Glossa: A Journal of General Linguistics*, *3*(1), 84. DOI: http://doi.org/10.5334/gjgl.149
Rice, K. 1989. *A Grammar of Slave (Dene)*. Berlin, Mouton de Gruyter.
Rice, K. 2000. *Morpheme order and semantic scope*. Cambridge: Cambridge University Press.
Rijkhoff, J. 2002. *The noun phrase*. Oxford University Press.
Ryan, K. M. 2010. “Variable affix order: grammar and learning.” *Language* 86 (4): 758-91.
https://dash.harvard.edu/bitstream/handle/1/34388150/ryan_variableaffixorder2010.pdf?sequence=1&isAllowed
Steddy, S. & V. Samek-Lodovici. 2011. “On the ungrammaticality of remnant movement in the derivation of Greenberg’s universal 20.” *Linguistic Inquiry* 42(3). 445–469. DOI: https://doi.org/10.1162/LING_a_00053
Steedman, M. 2011. “Greenberg’s 20th: The view from the long tail.” Manuscript, University of Edinburgh.
Stump, G. T. (2001). *Inflectional morphology*. Cambridge: Cambridge University Press.
TensorFlow: www.tensorflow.com
Wu, Y., M. Schuster, Zh. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun et al. 2016. “Google’s neural machine translation system: Bridging the gap between human and machine translation.” Technical Report, arXiv:1609.08144v2 [cs.CL]