Morphological productivity: A Bayesian modelling approach

Thesis
submitted in partial fulfillment of the requirements for the degree
Master of Science
in Cognitive Systems: Language, Learning, and Reasoning

Universität Potsdam
Faculty of Human Sciences
Department of Linguistics

Submitted by:	Elizabeth Pankratz (804865) born on July 14, 1994 in Edmonton, Alberta, Canada
First supervisor:	Prof. Dr. Shravan Vasishth
Second supervisor:	Jun.-Prof. Dr. Titus von der Malsburg

Potsdam, August 3, 2021

Statement on the originality of this work (Selbstständigkeitserklärung)

I hereby declare that this thesis has been composed solely by me and with no assistance from others beyond the sources acknowledged. All material taken directly or indirectly from external sources has been declared as such, and all references have been provided.

Further, I state that I am familiar with the guidelines for good scientific practice for students at the Universität Potsdam and the plagiarism policy from October 20, 2010 (*Richtlinie zur Sicherung guter wissenschaftlicher Praxis für Studierende an der Universität Potsdam (Plagiatsrichtlinie) vom 20. Oktober 2010*; `link`).

Elizabeth Pankratz
Potsdam, August 3, 2021

German abstract

Morphologische Produktivität wird in zahlreichen Studien der Wortbildungsforschung untersucht, aber die gebräuchlichen Maße weisen unerwünschte Abhängigkeiten von der Stichprobengröße auf. In dieser Masterarbeit schlage ich ein neues Produktivitätsmaß vor, das nach einem bestimmten Punkt von der Stichprobengröße unabhängig ist, nämlich die Shannonsche Entropie der Typenfrequenzverteilung eines Morphems. Der Grund für die Stichprobengrößenunabhängigkeit dieses Maßes ist, dass Typenfrequenzen ungefähr nach einem Potenzgesetz verteilt sind. Potenzgesetze besitzen die Eigenschaft der Selbstähnlichkeit: ihre Form ist in allen Größenordnungen gleich. Da Entropie die Form einer Typenfrequenzverteilung beschreibt, erreicht sie dementsprechend, wenn die Stichprobe ausreichend groß geworden ist, einen stabilen und interpretierbaren Wert.

Dazu behaupte ich, dass Entropie als Produktivitätsmaß konzeptuell gültig ist, weil sie in einer Bayesschen Regressionsanalyse die Tendenz zeigt, von Faktoren vorhergesagt zu werden, die Produktivität bekanntlich beeinflussen. Zum Schluss demonstriere ich die praktische Anwendung der Entropie anhand von zwei häufigen Fragestellungen in der Produktivitätsliteratur: zum einen dem Vergleich zwischen der synchronen Produktivität von mehreren Morphemen, und zum anderen dem diachronen Produktivitätswandel einzelner Morpheme.

English abstract

Measuring morphological productivity based on corpus data is a common goal for researchers of derivational morphology. However, the standard productivity measures all show undesirable dependencies on sample size. In this thesis, I propose a new measure of productivity that, having converged, does not show this dependency. This new measure is the Shannon entropy of a morpheme's type frequency distribution. The entropy of a type frequency distribution does not depend on sample size because type frequencies approximately follow a power law, and power laws are self-similar: they have the same shape, no matter the scale. Entropy summarises the shape of the type frequency distribution, so once the sample gets large enough, entropy stabilises at an interpretable value.

Further, I illustrate using a Bayesian linear regression model that entropy shows probable associations in the expected directions with factors believed to affect productivity. I also offer practical advice for how to use entropy to approach two common questions in the productivity literature: how the productivity of several morphemes compares synchronically, and how the productivity of individual morphemes changes diachronically.

Acknowledgements

Thank you, first and foremost, to my inspiring supervisors, Shravan Vasishth and Titus von der Malsburg, for teaching me more in the last two years than I thought I could learn. If I've become a better scientist, it's because of you.

Next, sometimes all that kept me going through this challenging time was the support of some spectacular people, especially my family, the Naive Baes, and the Saturday Dinner Gang. I want to thank several people in particular who helped this thesis become what it is: Rodrigo and Roland, for your conceptual input, and David, Lena, Nick, and Wellesley, for your eagle-eyed feedback on an earlier version of this text.

Finally, I gratefully acknowledge the financial support I received during this MSc from the Universitätsstipendium Potsdam and the Sir James Lougheed Award of Distinction.

Contents

List of Figures

List of Tables

1 Introduction

The main contribution of this work is a corpus-based, eventually sample-size-independent measure of morphological productivity. When a morpheme is productive, it can be readily used to form new words (Bauer 2001: 10, Dal & Namer 2016: 70). What makes productivity interesting is that it is not the same for all morphemes. Consider, for example, the English nominalisers *-ness*, as in *goodness*, and *-ity*, as in *serenity*. We, as competent users of English, have a sense that, if we needed to build a new noun to describe some property, say the property of being grotesque, we would probably reach for *-ness* to build *grotesqueness* before we would reach for *-ity* to build *grotesquity*. This generalisation holds for a large number of English adjectives. In other words, our sense is that *-ness* is more productive than *-ity*.

One way to learn about this kind of linguistic intuition is to look at usage data contained in corpora (Bybee & Beckner 2009, Diessel 2017, Milin et al. 2016), because the way we use morphologically complex words should reflect our knowledge of productivity. The idea of using corpora as an avenue to study morphological productivity goes back to Baayen (1989), and since then, a substantial body of work has arisen that proposes and applies various corpus-based measures. However, these measures all suffer from a weakness that is very common among values computed using corpus data: the measures show a dependence on sample size (Tweedie & Baayen 1998, Pankratz 2019). That is, the scores that these measures produce vary systematically based on the number of tokens in the sample used to compute them.

This sample size dependence affects studies of productivity in particular because of the types of questions that those studies tend to ask. Very often, their aim is to contrast the productivity of several morphemes at a single point in time (e.g., Bauer 2001, Doerfert 1994, Hay & Baayen 2003, Gaeta & Ricca 2006, Plag 1999, Plag et al. 1999, Schneider-Wiejowski 2011), or to track the productivity of one morpheme across several points in time (e.g., Bauer 2001, Cowie 1999, Cowie & Dalton-Puffer 2002, Demske 2001, Hartmann 2016, Kempf 2016, Scherer 2005, 2007). In short, their aim is to compare: either morpheme to morpheme, or time t to $t + 1$. But when a corpus contains more tokens of one morpheme than another, or when more text has survived from recent centuries than from more distant ones, then sample sizes will differ. And when sample sizes differ, that difference will affect the productivity measures calculated for those samples, confounding the measures and making their comparison uninterpretable (Lüdeling 2009, Zeldes 2012).

This problem can be avoided, for instance by trimming all samples to the largest common size (Zeldes 2012: 64) or by drawing many identically-sized subsamples from variously-sized initial samples (Hein & Brunner 2020: 35). But the problem can also

be eliminated entirely by applying a measure of productivity that does not depend on sample size. In what follows, I will argue that such a measure is given by the Shannon entropy of a sample's type frequency distribution. Entropy has three properties that make it desirable as a measure of productivity (and that form the outline of the rest of this thesis):

1. Interpretability: Once the sample is large enough, entropy stabilises at different values for differently productive morphemes, allowing us to assess relative productivity by comparing entropy values (Section 2).
2. Validity: Predicting entropy using factors that are known to affect productivity yields tendencies in the theoretically expected directions, as shown by a Bayesian linear regression model (Section 3).
3. Applicability: Entropy can be used to compare the productivity of multiple morphemes synchronically and, in some circumstances, to track how the productivity of individual morphemes changes diachronically (Section 4).

2 Interpretability: Entropy as a measure of productivity

Counting the number of times that each distinct word—that is, each type—appears in a sample gives that sample's type frequency distribution. In this thesis, I propose that drawing a sample of words all derived with a given morpheme, getting that sample's type frequency distribution, and then computing the Shannon entropy of that distribution offers a sample-size-independent way of measuring that morpheme's productivity. This assertion arises naturally from the reasoning that I will lay out in this section.

In brief: type frequencies can be described mathematically as a power law (Zipf 1949), and power laws have a property called self-similarity: no matter the scale at which you view them, the shape of the curve is the same (Mitchell 2009). This idea will be explored in more detail in Section 2.1. That power law distributions are self-similar is the key insight here. Because the shape of the distribution does not change, if we can find a way to summarise that shape, then the summary measure should be the same no matter the scale of the distribution—in other words, no matter the size of the sample.

One way to summarise the shape of a distribution is to use entropy. And conveniently, entropy values line up with intuitions about different morphemes' relative productivity; Section 2.2 will illustrate this and discuss why it is so.

While I am not the first to observe that entropy varies between differently productive morphemes (see, e.g., Hay & Baayen 2003: 15, Hein & Brunner 2020: 34), to my knowledge I am the first to suggest that entropy be used for more than just describing type frequency distributions and to propose that it actually be used to measure productivity.

It should also be noted that my proposal runs counter to a comment in Evert & Baroni (2020): the authors state that entropy is not recommended as a productivity measure because it is "not a reliable estimator of population entropy" (p. 75), presumably due to its well-known negative bias (Bonachela et al. 2008, Roulston 1999). I would dispel any concerns raised by this comment with two points. First, my aim here is not to estimate the entropy of the population of all derivations, but rather to use entropy to summarise the given samples only. I make no claims about how those estimates might be extrapolated to a population, so I am not concerned about the entropy estimator's negative bias. Second, the systematic sample size dependence that other productivity measures exhibit is more directly problematic for quantifying productivity than is entropy's negative bias. This dependence will be illustrated for three commonly-used measures in Section 2.3. Entropy's sample size independence sets it apart from those measures, a distinction that makes it, despite any weaknesses it may have, the most appropriate tool for quantifying productivity.

Nevertheless, there are some limitations that apply to all corpus-based measures, including entropy, and I conclude in Section 2.4 with a discussion of issues that still remain unsolved.

2.1 Type frequency distributions and self-similarity

When counting the number of times each word appears in a sample of natural language data (i.e., counting the number of tokens for each type), one will find that a small handful of types occur very frequently, a few more appear with middling frequency, and most show up only once or twice. Take, as a toy example, the previous sentence. Figure 1 shows the frequency distribution of the lemmatised types in that sentence. Even though the sample is small, it still follows this pattern.[1]

Famously, Zipf (1949) proposed a mathematical law to describe the distribution of type frequencies. This expression has come to be known as Zipf's Law, and it is defined as

$$f(w) = \frac{C}{r(w)}, \tag{1}$$

where $f(w)$ stands for the frequency of type w, $r(w)$ for the rank of type w (the most frequent type has rank 1, the second-most-frequent has rank 2, etc.), and C is a constant that is approximately equal to the frequency of the type with rank 1 (Baroni 2009: 813).

[1] Code for this and all other analyses in this thesis can be found on GitHub at `https://github.com/epankratz/entropy-productivity-thesis`.

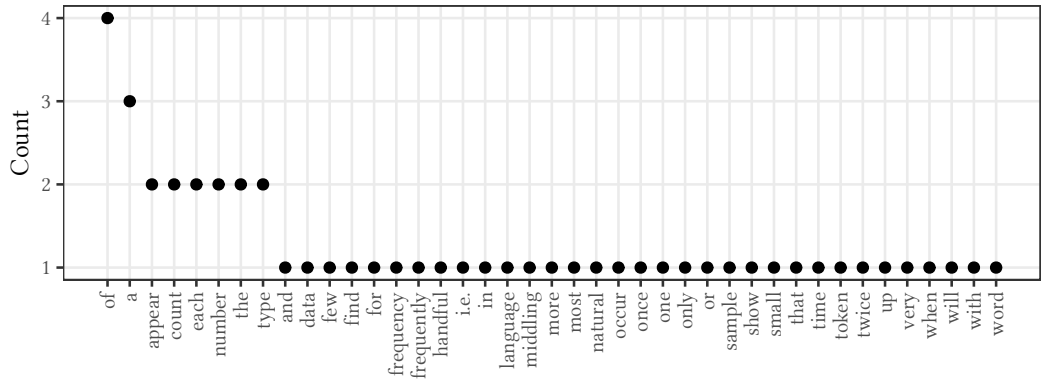

Figure 1: The type frequency distribution of the first sentence in Section 2.1

This law states that the frequency of the second-most-frequent type in a sample is about half the frequency of the most frequent type; the third-most-frequent type is about one-third as frequent as the most frequent one; and so on.

Generally, an additional parameter a is included in statements of Zipf's Law (Adamic 2000, Baayen 2001, Baroni 2009, Zeldes 2012).

$$f(w) = \frac{C}{r(w)^a} \tag{2}$$

If $a = 1$, as Zipf assumed, then Equation 2 is equivalent to Equation 1.[2] However, including this parameter allows Zipf's Law to be expressed as a power law, a type of expression that in general has the form x^d (Mitchell 2009: 245). Here, $x = r(w)$ and $d = -a$.

$$\begin{aligned} f(w) &= \frac{C}{r(w)^a} \\ &= C \cdot r(w)^{-a} \\ &\propto r(w)^{-a} \end{aligned}$$

That Zipf's Law of type frequencies can be expressed as a power law is relevant because of a particular property that power laws have, namely self-similarity: regardless of the scale at which one views the curve of a power law, it has the same shape (Mitchell 2009: 243). This is illustrated by the three plots in Figure 2. They show the curve described by Zipf's Law on three different orders of magnitude (inspired by Mitchell 2009: Figure 15.6). Such curves can also be described as being "scale-free".[3]

[2] For natural language data, a generally ranges between about 0.7 and 1.2 (Biemann & Quasthoff 2009: 2).

[3] For an exploration of the connection between power laws and fractals, which helps illuminate why power laws are scale-free, see Chapters 7 and 17 of Mitchell (2009).

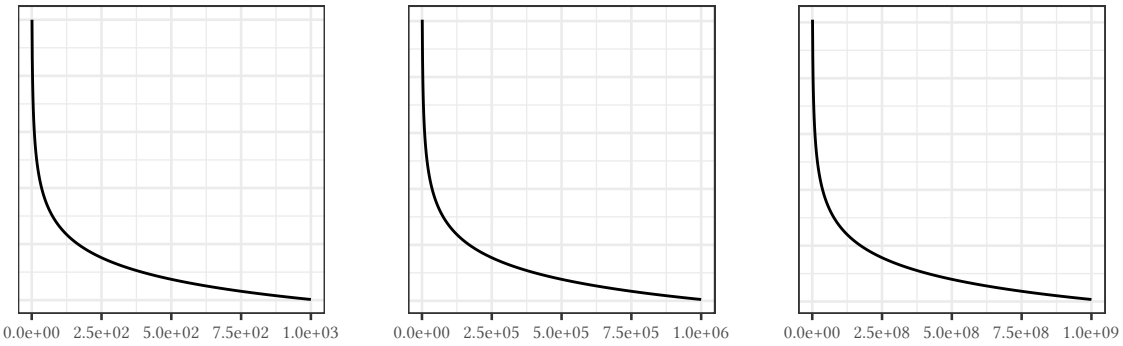

Figure 2: Zipf's Law (here shown with $C = 1, a = 0.1$) yields self-similar curves at three different scales

Like most real-life examples of fractal-like structures such as coastlines, crystals, and snowflakes, type frequency distributions are not perfectly scale-free. For one, type frequencies are discrete and cannot be infinitely small. For another, as observed by Mandelbrot (1953), the high- and low-frequency extremes of these distributions diverge somewhat from the frequencies that Zipf's Law predicts. Nevertheless, recognising that type frequency distributions are near-self-similar allows us to find an approximately scale-free—i.e., an approximately sample-size-independent—measure that we can use to describe morphological productivity.

2.2 Entropy and (eventual) sample size independence

The Shannon entropy for the probability distribution p in which each event i has the probability p_i is given by Equation 3 (Shannon 1948).

$$H(p) = -\sum_i p_i \log_2(p_i) \tag{3}$$

Conceptually, entropy can be described in several ways. If we take p as the probability distribution of a random variable, then the Shannon entropy gives the expected number of bits required to encode the information of an event from that random variable (Goodfellow et al. 2016: 71). Put differently, entropy measures the uncertainty in p, which has to do with the number of ways that the distribution p could have been generated (see McElreath 2020: 300–303 for an illustration). However, most simplistically, entropy indicates how "spread out" p is; the more spread-out the distribution, the higher the entropy.

And the spread of a distribution translates directly to productivity. We would expect a higher-productivity morpheme to have a more spread-out type frequency distribution. This is because that distribution will not only contain the usual high-frequency types, but

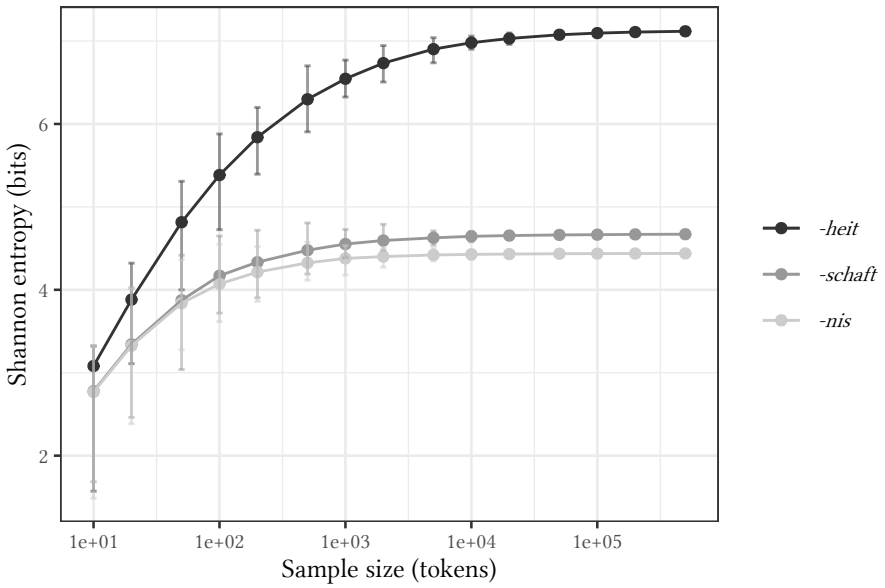

Figure 3: Shannon entropy as a function of sample size; points indicate the mean of 500 bootstrapped samples and error bars give the full range of bootstrapped estimates

will also have a long tail created by the many low-frequency types that result from the morpheme being used to coin new words. In contrast, a lower-productivity morpheme should have a less spread-out distribution that consists mainly of high-frequency types, since any new words that would appear with lower frequencies are not being coined as much (if at all). And because entropy is higher for more spread-out distributions, we expect that higher productivity is indicated by a larger value of entropy, and lower productivity by a smaller value.

To bring the ideas of self-similarity and entropy together: because type frequencies approximately follow a power law, the shape of their distribution should be near-scale-free; it should not change as sample size changes. Thus, when we summarise the distribution's shape using entropy, the values we obtain should also be independent of sample size.

We can tell if this is the case by drawing samples with a range of sizes, counting the types in those samples, and computing the Shannon entropy of each type frequency distribution. If the entropy is stable as sample sizes increase, then the idea of self-similarity will be borne out.

Figure 3 shows, for three example suffixes in German, the Shannon entropy of their type frequency distributions as a function of sample size. Because type frequency distributions are not perfectly scale-free, as noted above, entropy is not yet constant at smaller sample sizes. But, crucially, once sample size increases enough, entropy stabilises. And more than that, the value at which it stabilises is indeed different for each morpheme,

and the ordering of these values reflects our sense about how productive these suffixes are in relation to one another (namely that *-heit* is more productive than *-schaft*, which is more productive than *-nis*).

It is also noteworthy that, the higher the ultimate entropy is—i.e., the more productive the morpheme—the longer the curve takes to stabilise. This makes sense; the more detail a distribution contains, the longer it takes for that detail to fully emerge. A mathematical law that approximates when each curve stabilises will be discussed below in Section 5.2.2.

Before moving on, a comment is in order about how the data in Figure 3 came to be, since this method will be used repeatedly in what is to follow. The data for each sample size was bootstrapped, that is, randomly sampled with replacement (Efron 1979), from three original samples that consist of all of the derivations that contain *-heit*, *-schaft*, and *-nis* in the one-billion-token web corpus DECOW16A-NANO (Schäfer & Bildhauer 2012, Schäfer 2015). *-heit* appears 1, 706, 488 times, *-nis* 589, 279 times, and *-schaft* 795, 870 times. Five hundred bootstrapped samples were drawn for each sample size.

Bootstrapping these smaller samples may be a controversial choice. Random sampling overlooks the way that types are actually distributed in texts. Rather than occurring randomly, types tend to cluster based on texts' thematic and communicative needs, and this results in stretches of text that contain many relevant types, followed by desert-like expanses containing none—a situation referred to as "underdispersion" (Zeldes 2012: 84, Evert 2004: 420–421). Nevertheless, pretending that types are generated at random is a very common simplifying assumption when modelling type frequencies (Baayen 2001, Evert 2004), so I make it here, too.

The second point about bootstrapping is that it involves sampling *with* replacement (Efron 1979: 3). In other words, tokens randomly drawn for one subsample are returned to the pool and may be drawn again for that same subsample. Baroni & Evert (2016: 54) state, however, that sampling with replacement is inappropriate for type-token distributions because the number of types is underestimated, and that samples should be drawn *without* replacement instead. I have repeated all analyses shown here using sampling without replacement, as they suggest, and the pattern of results does not change (see Appendix A for more discussion and these supplementary analyses). I use classic bootstrapping here because sampling with replacement more closely resembles the process of drawing new samples from the population.

To sum up, what we have seen so far is the following: because type frequency distributions are near-scale-free, the entropy of those distributions eventually approaches a stable and interpretable asymptote. To show that this behaviour is unique among productivity measures that are commonly in use today, the next section illustrates the dependencies on sample size of three of those measures.

2.3 Comparison to standard productivity measures

The measures that this section will illustrate are type count, potential productivity $\mathscr{P}$ (Baayen 2001, 2009), and the parameter S from a finite Zipf-Mandelbrot model (Evert 2004). These three measures each shed light on different aspects of what it means for a suffix to be productive, according to Zeldes (2012: 94), who recommends these three measures (along with a morpheme's overall frequency) as the main ones that should be applied to evaluate a morpheme's productivity.

Type count and $\mathscr{P}$ are, in essence, summary statistics of samples containing tokens derived with a particular morpheme, while S is a parameter of a statistical model computed on the basis of such a sample. It is well-known that type count and $\mathscr{P}$ depend on sample size (see, e.g., Lüdeling 2009, Zeldes 2012), and Pankratz (2019) showed that this is also the case for S. I wish to emphasise at this point that these measures themselves are not wrong or unsuitable for measuring productivity; they are perfectly valid, but only under the specific condition that all sample sizes are the same.

Figure 4 shows the behaviour of each of these measures as sample sizes increase, produced using the same bootstrapping method described above. These plots clearly show that none of these three measures exhibit a tendency to stabilise at an interpretable value the way that entropy does.

So that the contrast to the entropy-based approach can be better understood, I briefly describe each of these three measures here. For more discussion, see Zeldes (2012: Chapter 3), and for mathematical details, see Baayen (2001).

2.3.1 Type count

Intuitively, a morpheme's productivity is reflected in the number of types that have been formed using that morpheme (Bauer 2001: 48–49, Kempf 2016: 79, Zeldes 2012: 49). A more productive morpheme will generally form a greater number of distinct words than a less productive one will. Thus, the larger the type count for a given morpheme, the more productive it is understood to have been.

For example, in one of the 500,000-token bootstrapped samples used in Figures 3 and 4, *-heit* has 3,785 distinct types, while *-schaft* has 849, and *-nis* 306. Because all of these counts come from samples that are the same size, the counts are comparable. We would say that, given these samples, *-heit* has produced more types and thus has (at least historically) been more productive than *-schaft* and *-nis*.

Differences in token counts are common, however. Imagine a situation in which we only have 10,000 tokens for *-heit*, compared to 500,000 each for *-schaft* and *-nis* (this example is overly dramatised, but it illustrates the point). *-heit* now only has 877 types, placing it on a par with *-schaft*. This is because the smaller sample size for *-heit* reduces

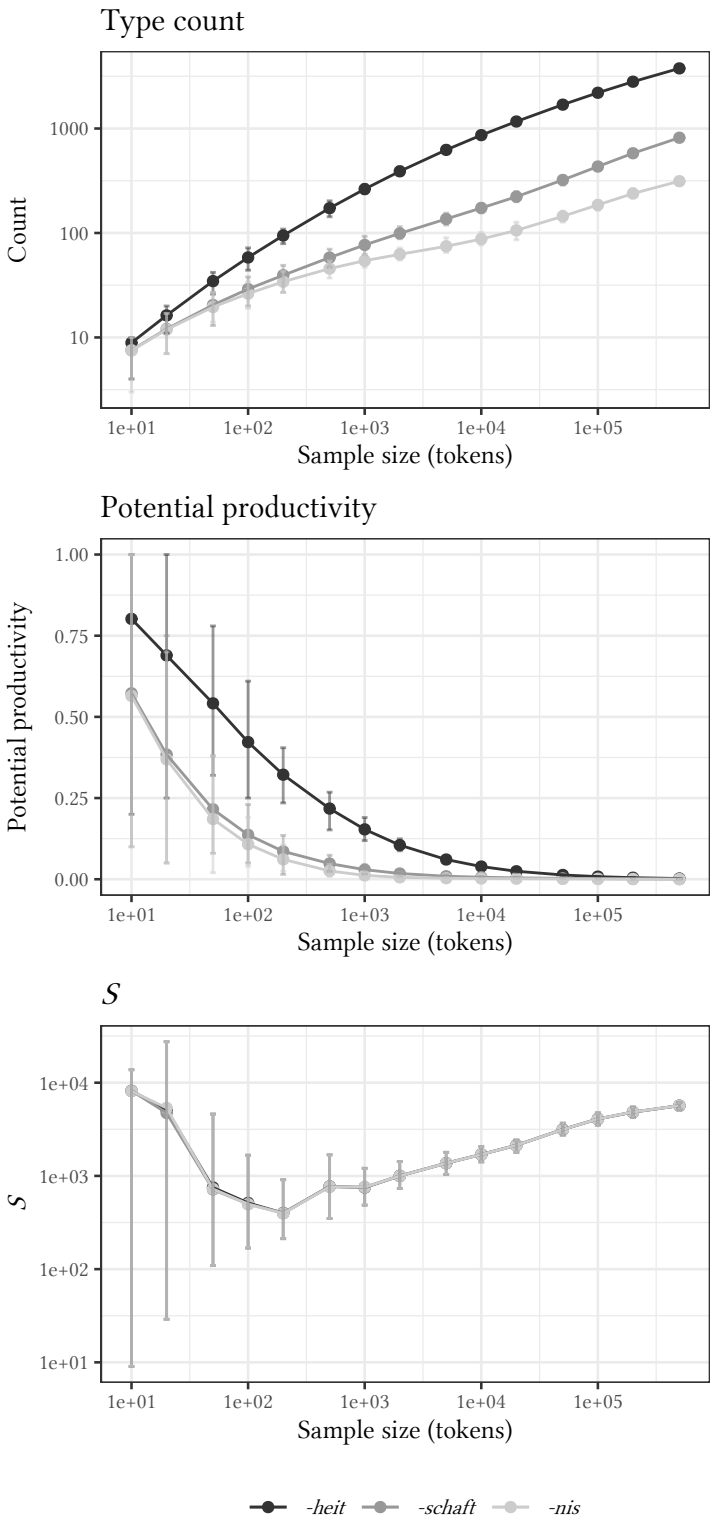

Figure 4: Three standard productivity measures as a function of sample size; points indicate the mean of 500 bootstrapped samples and error bars represent the full range of estimates, except for in S, where extreme outliers were removed (defined as more than 1.5 times the interquartile range away from the first and third quartiles; see Section 2.3.3)

the number of types that the sample can contain. Any conclusion about relative productivity based on type counts from differently-sized samples could well be confounded by the concomitant increase of sample size and type count visualised in the top-most plot in Figure 4. Thus, these numbers cannot be compared.

Type count is a rather naïve measure of productivity. What it does not consider is that there is an informative difference between the *kinds* of types that a morpheme can create. For example, if we focus on newly-coined types, i.e., neologisms, we can identify when a morpheme was historically very productive but no longer is, compared to a morpheme that is still regularly used to coin new words today. The type counts for both of these situations would be large. But the next measure, which considers the rate at which new types arise, should be able to distinguish them.

2.3.2 Potential productivity

In order to see how frequently new types are being coined, we need to identify how many neologisms appear with a given suffix. We can approximate the neologisms by identifying the types that only appear once in a sample, the so-called *hapax legomena* (Classical Greek for "said once"). Using hapaxes in this way has been standard practice since Baayen (1989).

The basic idea behind hapax legomena as a stand-in for neologisms is that, if a language user has just coined a new word, then that word will be brand new, so it will only appear once in the sample. Of course, this approximation is not perfect. Some hapaxes may be established words that just happen to only appear once in the given sample, so the assumption is that true neologisms are a subset of hapaxes (Zeldes 2012: 60). However, it is more likely that they are two partially intersecting sets, since there are also situations in which neologisms may appear twice or more in a sample. For instance, when a language user coins a new word, they probably do so because they wish to discuss some concept or idea. And in such a discussion, that concept or idea may be mentioned more than once, disqualifying it from the set of hapaxes. Additionally, the same word may be coined separately by two individuals, and if both coinages appear in one sample, then on the hapax-based approximation, that type will also not be considered a neologism. All in all, despite the imperfect mapping, hapaxes are the commonly-accepted stand-in for neologisms because of the easy operationalisation they provide (Zeldes 2012: 60–61).

The number of hapax legomena alone is generally not used as an indicator of productivity. Rather, following Baayen (2001, 2009), it is generally incorporated into other measures, the most widely-used of which is called "potential productivity".[4] Potential

[4] This measure has also variously been called "productivity in the narrow sense" or "the category-conditioned degree of productivity" (Scherer 2005, Hay & Baayen 2003, Baayen 2009).

productivity is denoted as $\mathscr{P}$ and computed as shown in Equation 4, where h represents the number of hapax legomena in a sample and n the sample size in tokens.

$$\mathscr{P} = \frac{h}{n} \tag{4}$$

$\mathscr{P}$ can be interpreted as the probability that the next token encountered will be a hapax legomenon. In the words of Zeldes (2012: 64): "the basic idea is that at every point within a growing sample, [morphological] processes have an inherent probability of producing previously unseen forms, and that that probability decreases the more we have seen of a process's output".

The decrease in probability that Zeldes mentions is the way in which $\mathscr{P}$ depends on sample size. This dependency is apparent from Equation 4, since the sample size n is the denominator of the fraction. As sample size increases, i.e., as the denominator gets larger, $\mathscr{P}$ approaches zero (see the middle plot in Figure 4). Because of this dependency, the same caveats we saw for type count apply: $\mathscr{P}$ values can only be sensibly compared when the samples they are computed on are the same size (Lüdeling 2009).

As mentioned above, $\mathscr{P}$ provides a more nuanced view of a morpheme's productivity than type count alone. However, as straightforward summary statistics, they are both unable to predict the behaviour of a morpheme beyond the sample sizes available. The next and final measure we will consider extrapolates from the observed sample to a sample of arbitrary size, predicting the ultimate number of types that that morpheme is likely to generate.

2.3.3 *S*

Above I gave the number of types for *-heit*, *-schaft*, and *-nis* in samples of 500,000 tokens. Another way of describing those counts is as the number of types when all of the tokens in the sample have been "seen". Using this framing of tokens seen, we can now ask: starting from zero, as we see more and more tokens from our sample, how many distinct types do we encounter, and how quickly? Plotting the type count as a function of number of tokens seen results in the vocabulary growth curves shown in Figure 5 (the "vocabulary" of a sample is the set of distinct types in that sample; Baayen 2009, Baroni 2009).

A vocabulary growth curve has a characteristic shape. Initially, it rises fairly steeply. This is because, when only a few tokens have been seen, most of them will represent new types. However, the growth of the curve becomes more gradual as more and more tokens are encountered, and eventually, if all types in the sample have been seen, the curve evens out.

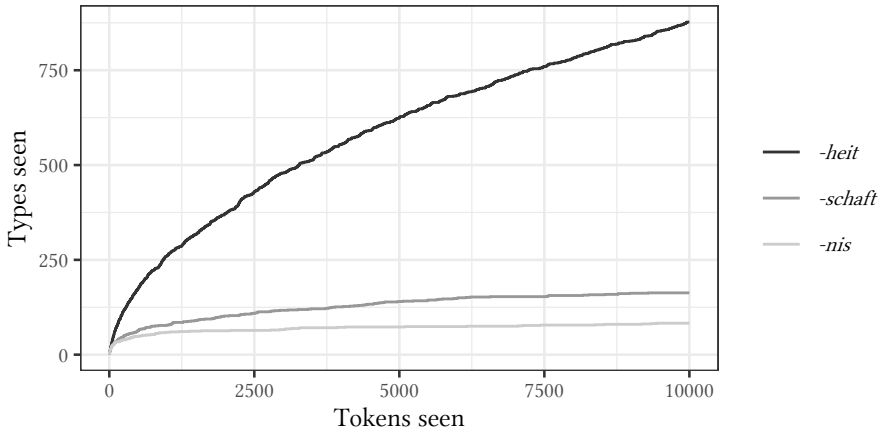

Figure 5: Vocabulary growth curves from a 10,000-token sample

Incidentally, $\mathscr{P}$ has an interpretation related to the vocabulary growth curve: it is the slope of the curve at its endpoint, i.e., at the maximum number of tokens observed (Baayen 2001, Zeldes 2012). $\mathscr{P}$ starting high but tending to zero in the limit corresponds to the vocabulary growth curve's initial steep rise but eventual flattening-out.

The point on the vertical axis where the vocabulary growth curve evens out, even if that point is not yet reached by a given sample, represents the ultimate number of types that a morphological process will produce: the upper limit of its productivity. This is the quantity denoted by S.[5] This parameter is estimated using what Baayen (2001) calls an LNRE model (for "large number of rare events"), specifically a finite Zipf-Mandelbrot model (Evert 2004). The details of this model are peripheral and I will not discuss them here; see Pankratz (2019: Section 4.4) for a high-level conceptual overview and Evert (2004) for the mathematical detail. For now, it is enough to know that a finite Zipf-Mandelbrot model looks at a sample of derived words and, based on this sample, predicts S, the maximum number of types that the given derivational morpheme is expected to produce. The greater the value of S, the higher the productivity of that morpheme.

Although S has been claimed to be sample-size-independent for sufficiently large samples (Zeldes 2012: 93), Pankratz (2019: 35–37) showed that this is not the case. S is erratic at small sample sizes and shows an upward trend at large ones (see the third plot in Figure 4). Although S appears to even off as sample size increases, this is probably an effect of the bootstrapping having been done with replacement; as Appendix A shows, when S is computed on samples drawn without replacement (which is, in fact, what Baroni & Evert 2016 advocate), the curve continues constantly upward. Thus, although the interpretation of S is in principle uncoupled from sample size, the estimation of S is not.

[5] The notation of this quantity as S comes from the study of population growth in animals, where S stands for "species" (Zeldes 2012: 76).

Further factors make the interpretation of S difficult. For one, it sometimes produces unreasonably large estimates, on the order of 10^{50} and beyond (Pankratz 2019: 35, Schneider-Wiejowski 2011: 188). These outliers were removed for the visualisation in Figure 4. And for another, the estimates for the three suffixes illustrated here are not distinguishable; the lines overlap. It is difficult to learn from this measure how the expected productivity of these suffixes should differ (and, considering the rapid taper in the vocabulary growth curves of *-schaft* and *-nis* in Figure 5, it is surprising that the ultimate number of types for these suffixes is predicted to end up in the thousands).

To summarise, we have seen that the three most-recommended productivity measures are all dependent on sample size, meaning that they cannot sensibly be compared when sample sizes vary. This stands in contrast to an entropy-based approach. However, it would be unfair to these measures to gloss over an important conceptual limitation that entropy shares with type count, $\mathscr{P}$, and S, and indeed with all productivity measures that draw exclusively on data from corpora.

2.4 General limitations of corpus-based productivity measures

Morphological productivity is a complex, multifaceted phenomenon—this is why it so captivates morphologists. But it also means that there are aspects of productivity that cannot be captured with corpus data alone. For example, there are some morphological processes that are very productive, but within such a restricted domain of application that the number of types they produce will be very limited (Riehemann 1998: 60). In German, the prefix *ein-* can be freely applied to any verb that means 'to sleep', resulting in a verb meaning 'to fall asleep': *einschlafen*, of course, but also *einschlummern*, *einnicken*, *einpennen*, and so on. And in French, the suffix *-u* can create adjectives from any noun denoting a body part, e.g., deriving from *barbe* 'beard' the adjective *barbu* 'bearded' (Dal & Namer 2016: 73). In both of these cases, the domain of application is so narrow—verbs meaning 'to sleep' and nouns for body parts—that the maximum number of types created by these derivational processes will be very small. Any measure that relies on frequencies, including entropy, will not reflect the immense productivity of these morphemes within their very limited domain.

More broadly, consider the flow chart in Figure 6. This figure is my translation and reproduction of the original German version in Kempf (2016: Figure 17). It illustrates the great web of factors that influence a morpheme's productivity (informed partially by Kempf's construction-morphology perspective). The issue with *ein-* and *-u* falls under the factor "Inventory of suitable bases", a factor that modulates a morpheme's productivity beyond the grasp of current corpus-based approaches. "Need for new words", too, is something that cannot be so easily quantified. We must acknowledge that the corpus-

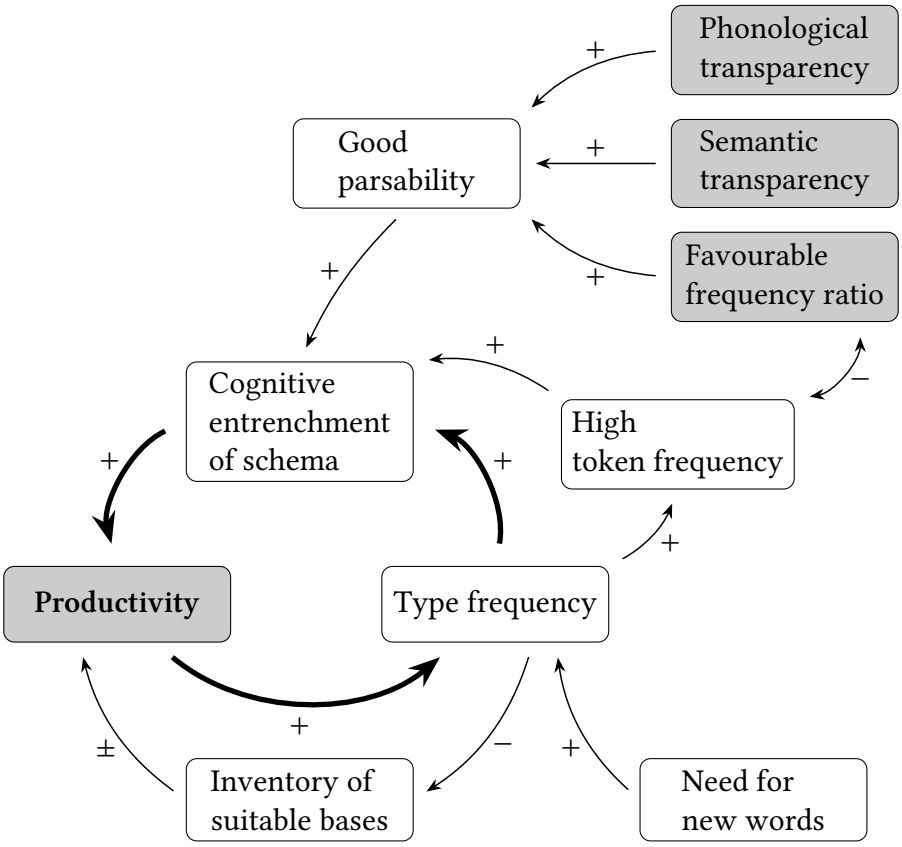

Figure 6: A translated reproduction of Figure 17 in Kempf (2016) showing factors affecting productivity and the direction of their hypothesised influences; shading represents inclusion in the present study

based approach to productivity can only fill in part of this very nuanced picture. I will return to this point in the conclusion below.

In her study, Kempf (2016) only quantified type and token frequencies and considered the rest at a qualitative level. In the next section, I will operationalise and quantify the factors presented in Figure 6 in grey and show how they affect entropy.

3 Validity: Evaluating factors that influence productivity

If factors that are known to affect morphological productivity are also associated with entropy in the predicted way, then conceptually, entropy must be picking up on some aspects of what it means for a morpheme to be productive. This would strengthen the position that entropy can suitably measure productivity. Thus, this section aims to validate entropy as a productivity measure by assessing to what extent entropy can be predicted using the three factors in Figure 6 that influence parsability: a favourable frequency ratio, semantic transparency, and phonological transparency.

I have chosen these particular factors to evaluate because, according to Figure 6, they are the ones that can be most easily interpreted as causes of productivity rather than effects of it. If we read that flow chart as if it were a causal graph (see, e.g., McElreath 2020: Chapter 6), then the causal links between the parsability factors and productivity are, on the whole, much less circular than the others. Further, the factors "Inventory of suitable bases" and "Need for new words" cannot be easily quantified using the resources available, and "Cognitive entrenchment of schema" is specific to the theoretical framework that Kempf assumes. That leaves the three factors that influence parsability.

To study these effects, I have sampled tokens formed by 35 German nominalising suffixes from DECOW16B (a nearly-17-billion-token web corpus, of which DECOW16A-NANO is a subset). Details about how these suffixes were chosen and how the samples were created can be found in Appendix B.2.

In what follows, I explain the idea behind each one of these factors and outline how I operationalised them (Section 3.1). Then, I use the factors as predictors in a Bayesian linear regression model that estimates their effect on entropy (Section 3.2). I will show that, although the model is not fully certain about the direction of the effects, in each case it still places most of the posterior probability mass in the hypothesised directions, a result that I take as a tentative validation of entropy as a measure of productivity.

3.1 Operationalising three factors

3.1.1 Frequency ratio of base and derivation

In order to be productive, a derivational morpheme must be identifiable to language users as an element that can be used in word formation. One factor that affects a morpheme's identifiability is how frequently its derivations are used compared to the base words that those derived words build on. If the base is consistently much more frequent than the derivation, then upon encountering a derivation, language users will more easily be able to identify that it is an expansion on some existing form. On the other hand, if the derivation is much more frequent than the base, then its status as a derivation may not be as clear (Hay 2001, Hay & Baayen 2003). In this way, the identifiability of a morpheme—and thus its productivity—is affected by the relative frequency of base words to derivations using that morpheme.

Take, for example, the German suffix *-and*. In my data, it has a low base-derivation frequency ratio. Its most frequent type *Konfirmand* 'confirmee' (a person undergoing Christian confirmation) appears 926 times in DECOW16A-NANO, while the base verb *konfirmieren* 'to confirm' appears only 33 times. This suggests that, on the whole, the suffix *-and* is less distinct and thus probably less productive than a suffix like *-schaft*,

whose bases are, overall, much more frequent than its derivations (for instance, *Landschaft* 'landscape' appears 28,279 times, while its base *Land* 'country' appears nearly ten times as often with 247,182 tokens).

I counted how often in DECOW16A-NANO each derivation and its base appear.[6] I then computed the mean frequency ratio for each suffix using Equation 5, in which b_i represents the frequency of the i^{th} base and d_i the frequency of the i^{th} derivation in a sample of n tokens.

$$\frac{1}{n}\sum_{i=1}^{n}\log\left(\frac{b_i}{d_i}\right) \tag{5}$$

Taking the logarithm of the frequency ratio means that, if the base is more frequent than the derivation, the ratio's value is positive, while if the derivation is more frequent than the base, the value is negative.[7] Following the logic outlined above, we would predict a positive association between the log frequency ratio and productivity. Figure 7 shows the distribution of the mean log frequency ratios for the 35 suffixes in my dataset.

3.1.2 Semantic relatedness of base and derivation

The second factor I will consider is the semantic relationship between bases and derivations. If this relationship is predictable and regular, that is, if the same change in meaning happens every time a word is derived with the same morpheme, then language users will be more confident in what will happen semantically when they apply that morpheme to create a new word. Thus, a more regular semantic relationship should be associated with increased productivity (Kempf 2016: 74–76, Riehemann 1998: 60).

For example, consider the difference between the suffixes *-heit* and *-ling*. *-heit* nearly always builds nouns that describe some property or characteristic of people or objects, for example *ehrlich* 'honest' ~ *Ehrlichkeit* 'honesty' (Fleischer & Barz 2012: 212; *-keit* is an allomorph of *-heit*). In contrast, *-ling* is semantically much less predictable. It creates nouns for the result or subject of some process, such as *lehren* 'to teach' ~ *Lehrling* 'apprentice', but also agentive nouns, such as *flüchten* 'to flee' ~ *Flüchtling* 'refugee',

[6] At the time of writing, no resources exist that can automatically and reliably identify the bases of German derivations. I therefore wrote a rule-based Python module called `backformer` that takes a derivation and performs string manipulations to generate all its possible bases, one of which is generally correct. More detail about `backformer` is given in Appendix B.2.

[7] Approaching this factor as a continuous variable, as I have done here, is not the only option. Hay & Baayen (2002) use a computational model of morphological processing called Matcheck (Baayen & Schreuder 2000) to set a so-called "parsing line": a threshold on the frequency ratio beyond which, they argue, a derivation is likely to be decomposed into base and affix rather than processed as a single unit. Here I keep frequency ratio as a continuous variable for reasons both practical and methodological: practical, since I do not have access to Matcheck, and methodological, since I prefer to maintain all the information available in continuous variables, rather than binning them into potentially less informative categories.

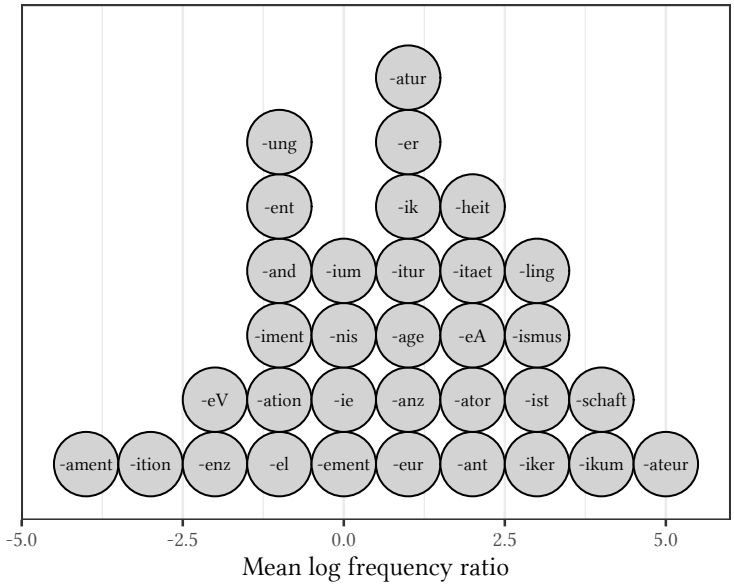

Figure 7: The mean log frequency ratio of each suffix; negative values indicate that the derivation is, on average, more frequent than the base, while positive values correspond to bases that are more frequent than their derivations

or pejorative terms like *feige* 'cowardly' ~ *Feigling* 'coward'—and much more besides (Fleischer & Barz 2012: 216–217). Because *-heit*'s semantic effect is much more regular—the meanings of its derivations are more closely related to the meanings of its bases—it is probably more productive than *-ling*.

I operationalised this factor as the probability that the base and the derivation are semantically related, and I made this choice for opportunistic reasons: the excellent resource DErivBase 2.0 (Zeller et al. 2014) already provides, for a very large number of morphologically related word pairs, these probabilities. They were learned by a binary classifier—specifically, a support vector machine with a radial basis function kernel—trained to predict semantic relatedness. The classifier used a set of 35 features that include distributional information like the cosine similarity between base and derivation embeddings, but also structural similarity at the level of the derivational rules (e.g., how frequent that rule is, how regularly it has been judged by human annotators to derive related forms, and so on; see Zeller et al. 2014: 1735 for details).

To compute the mean probability of semantic relatedness for each suffix, I identified all of the word pairs in the DErivBase data that consist of (i) a derivation containing the suffix of interest and (ii) a base that was generated for that derivation by my base identification system `backformer` (see Appendix B.2). I then took the mean of the probabilities provided by DErivBase. Following the logic above, a greater mean probability of semantic relatedness should be positively associated with productivity.

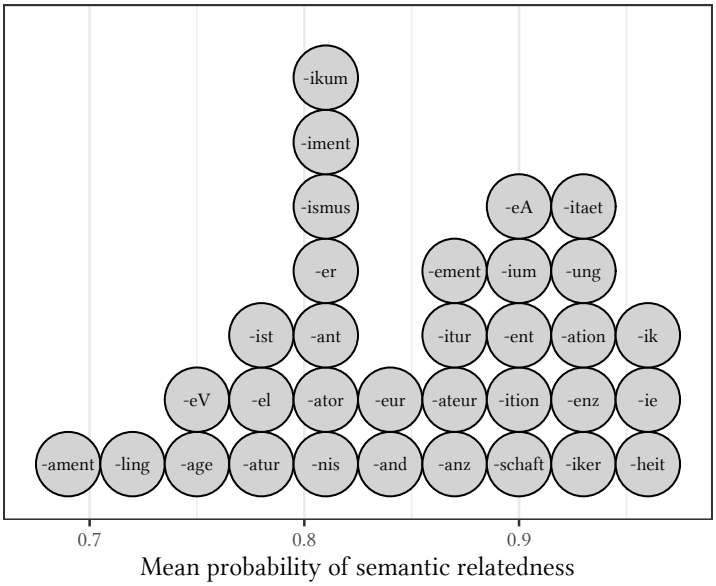

Figure 8: The mean probability of semantic relatedness for each suffix

Figure 8 shows the distribution of these scores for the 33 German suffixes that appear in DErivBase 2.0. Two suffixes were absent from that dataset, namely *-end* (as in *promovieren* 'to do a PhD' ~ *Promovend* 'doctoral candidate') and *-iteur* (as in *repetieren* 'to repeat' ~ *Repetiteur* 'rehearser, coach, tutor'); consequently, these two suffixes do not appear in the statistical analysis.

3.1.3 Phonotactic properties of the juncture of stem and affix

The final factor I will study calls upon phonological knowledge. Language users apply knowledge about phonotactic properties of words—that is, about what sounds may occur where—to segment continuous input into individual words (Saffran et al. 1996a,b). It is therefore plausible to expect that phonotactic knowledge also helps language users identify word-internal morpheme boundaries (Hay & Baayen 2003). A morpheme boundary will not be very distinct if the sound sequence that bridges it also frequently appears in monomorphemic words (i.e., simplexes). Take, for example, the sequence /ns/ in *insincere*; /ns/ often appears in English simplexes like *pansy*, *insert*, or *tinsel*. In contrast, if the bridging sound sequence is rare in simplexes, then the existence of the morpheme boundary is much clearer (e.g., /nh/ in *inhumane*, which appears in English simplexes much less, if at all; examples from Hay & Baayen 2003: 8). As these examples show, a single morpheme can have both high- and low-frequency junctures, but on the whole, the mean probability that the juncture appears in simplexes varies between morphemes and correlates with productivity (Hay & Baayen 2003). I thus use junctural phonotactics to operationalise Kempf's factor of "Phonological transparency".

To compute the mean probability that a morpheme juncture appears in simplexes, I created a list of German simplexes from DECOW16B (see Appendix C for details) and counted the frequency of each character bigram contained within them. In contrast to Hay & Baayen (2003), who looked at English and therefore had to use phonetic transcriptions to identify the sounds at morpheme boundaries, I could use character bigrams because of the liberty afforded by German's more transparent orthography.

That said, although German graphemes generally map fairly closely to phonemes, this approximation is not perfect. For example, the suffix *-schaft* begins with /ʃ/, while the bigram method will see <s> and assume it represents /s/. Also, the mappings differ slightly between German's core vocabulary and the more Latinate-leaning loan vocabulary. Future versions of this work could apply automatic tools for converting German orthography to the corresponding phonetic representations, e.g., `gramophone` (Würzner & Jurish 2015). However, in the interest of manageability, I accept the limitations of the orthography-based method for now.

One more difference between the procedure of Hay & Baayen (2003) and my own was that they computed phonotactics in a syllable-sensitive manner, conditioning the probability of each juncture on the relative syllable positions of the two segments. For example, in *investment*, the juncture consists of the coda /st/ and the onset /m/, so they computed that juncture's probability as

$$P(\text{coda-onset transition}) \times P(\text{/st.m/} \mid \text{coda-onset transition}),$$

using the definition of conditional probability, rearranged: $P(A \cap B) = P(B)P(A|B)$ (Hay & Baayen 2003: 11).

Taking syllable structure into account was possible because of their transcriptions, which contained syllable boundaries. Since I worked only with the surface-level orthography, I again simplified my analysis and ignored syllable structure.[8]

From the frequencies of character bigrams within simplexes, I computed the probability that each bigram appears in a simplex. I then matched these probabilities with the bigrams that appear at the morpheme junctures in my data, assigning bigrams that never appear in simplexes a probability of zero. Taking the mean of these probabilities for each suffix allows us to answer the question: how probable is it, on average, that a given suffix will create a juncture that is found in German simplexes? We would predict this probability to be negatively associated with a suffix's productivity.

[8] Tools do exist for automatic hyphenation of German words, e.g., `dehyphen` (`https://github.com/pd3f/dehyphen`) and `HypheNN-de` (`https://github.com/msiemens/HypheNN-de`), but these tools are intended for typesetting, not for identifying linguistically motivated syllable boundaries, so they are not applicable here.

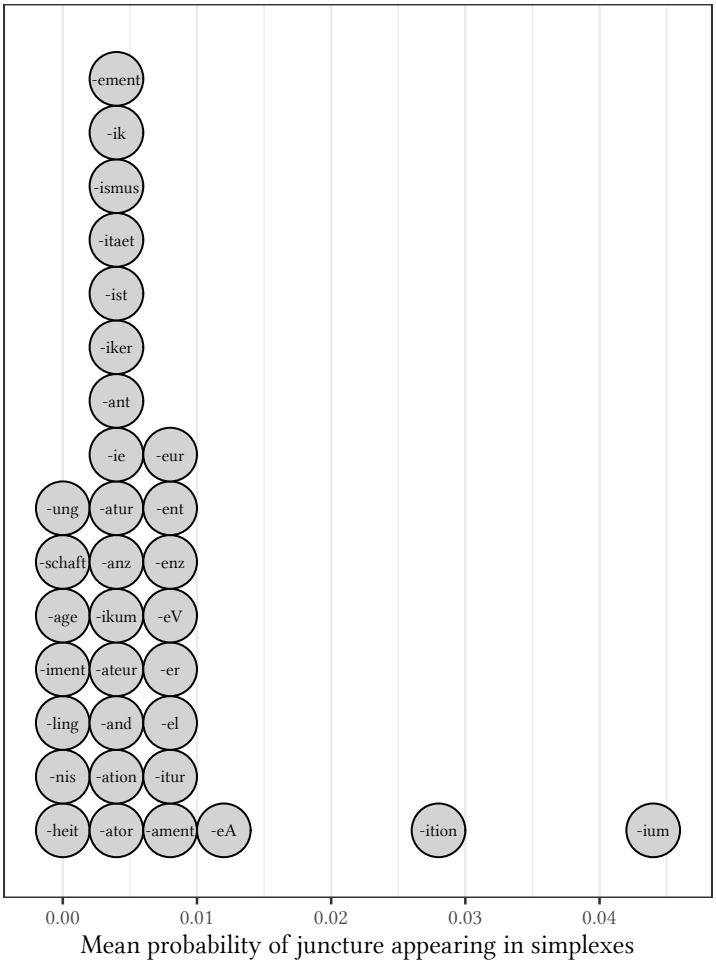

Figure 9: The mean probability that the character bigram at the morpheme juncture appears in German simplexes

Figure 9 shows the distribution of these mean probabilities. Table 1 summarises the hypotheses about how all three parsability factors are associated with productivity, and in Appendix D, all of the values for each predictor and suffix are given.

3.2 A Bayesian model of morphological productivity

Modelling entropy as a function of these three factors will help us identify whether entropy is a valid productivity measure. If the effects of these three factors go in the predicted directions, then it is probable that entropy reflects some aspect of morphological productivity. To estimate these effects, I fit a Bayesian linear regression model (Section 3.2.3), and to test the amount of evidence we have for each factor, I conducted a series of Bayes factor analyses (Section 3.2.4). Before exploring to the results, though, I briefly discuss how the data was preprocessed and how priors were determined.

Table 1: Hypothesised direction of association of each variable with productivity

Factor	Hypothesis
Frequency ratio	+
Semantic relatedness	+
Junctural phonotactics	−

3.2.1 Data preparation

The variables discussed above have very different scales: the values for junctural phonotactics are fractions of percentages, while the mean log frequency ratio ranges from about −4 to nearly 5. To simplify setting the priors and interpreting the results, I standardised all three variables to z-scores (Gelman & Hill 2007: 54): for observed value v, sample mean μ, and sample standard deviation σ,

$$z_v = \frac{v - \mu}{\sigma}.$$

Because we are interested in the effect on entropy of the three predictors, the parameters in the model that we will focus on are the β coefficients (see below). The z-score transformation means that these coefficients are interpreted not on their original scale, but rather in units of standard deviations.

As the pairs plot in Figure 10 shows, these variables are not strongly correlated with one another, so multicollinearity will not be an issue (McElreath 2020: Chapter 6.1).

3.2.2 The model

A Bayesian model is a statement about how we believe our data to have been generated (McElreath 2020: 28). Broadly speaking, it consists of several distribution functions that define how plausible certain values are. The distribution function that defines the plausibility of the outcome variable—that is, the data we have seen—is the likelihood. The functions that describe our assumptions or beliefs about plausible values for parameters within the model, such as the intercept and slope of a linear function, are the priors. Together, the product of the likelihood and the priors is proportional to the posterior probability of the model's parameters, conditional on the data we have observed; this is given by Bayes' theorem (McElreath 2020: Chapter 2.4, Nicenboim et al. 2021: Chapter 2.1). I will discuss the likelihood and the priors of my model in turn.

Because our outcome variable, entropy, must be positive, we require a likelihood function that only permits values greater than zero; for instance, a lognormal likelihood. If a variable y is lognormally distributed, then $\log(y)$ is normally distributed (McElreath 2020: 96, Nicenboim et al. 2021: Chapter 3.5.2).

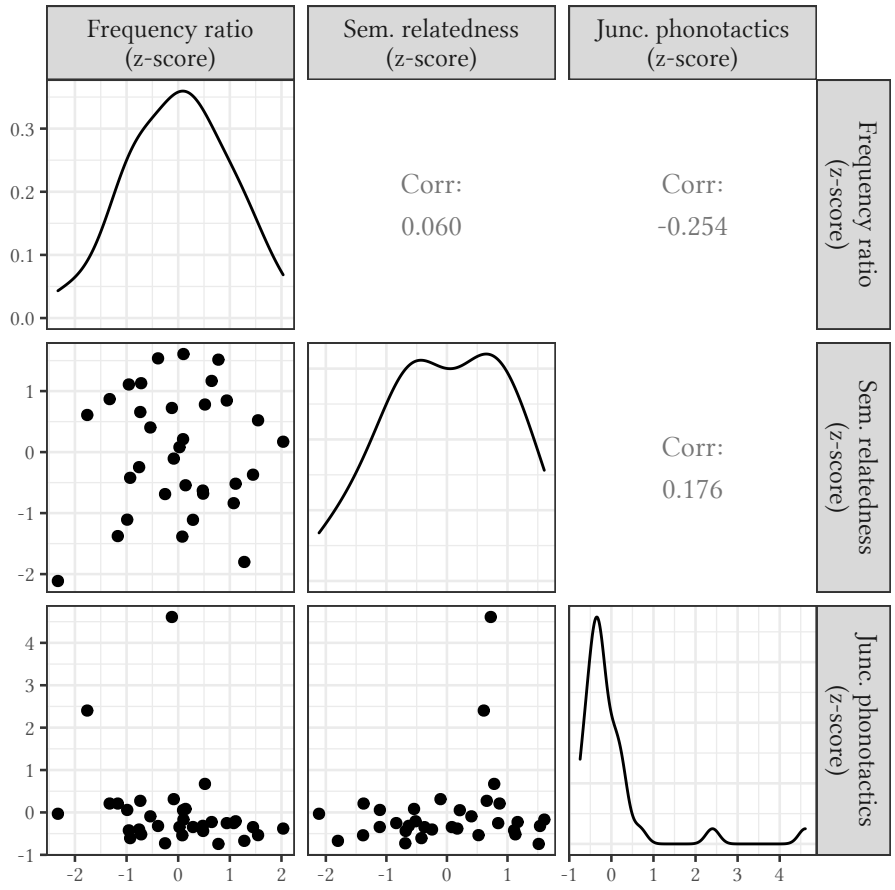

Figure 10: Only weak correlations between predictors

The lognormal distribution, $LogNormal(\mu, \sigma)$, has two parameters: the location μ and the scale σ (Nicenboim et al. 2021: Chapter 3.5.3). By replacing μ with a linear function that includes our three predictors of interest, we permit the value of entropy to be affected by these predictors. Thus we arrive at the likelihood of the model,

$$entropy_n \sim LogNormal(\alpha + (z\_fr_n \cdot \beta_{fr}) + (z\_sm_n \cdot \beta_{sm}) + (z\_jp_n \cdot \beta_{jp}), \sigma),$$

where n indexes each data point, and *z_fr* represents the z-score of the frequency ratio variable, *z_sm* that of semantic relatedness, and *z_jp* that of junctural phonotactics, each of which has a corresponding slope β. The intercept of this linear function—the value when all predictors are at their means of zero—is given by α.

The parameters α, β, and σ require priors. Priors encode assumptions that we make about the generative process behind the data. Since a Bayesian model is a generative model, we can simulate data from the priors to see whether those assumptions, in constellation, lead to sensible-looking data. Data simulated in this way is called the prior predictive distribution (Nicenboim et al. 2021: Chapter 3.2). I used prior predictive simulation to determine that the following are suitable priors for α and σ.

$$\alpha \sim Normal(1, 0.5)$$
$$\sigma \sim Normal_+(0, 0.3)$$

Note that, because of the lognormal likelihood, the parameters used in these priors are on the log scale. This, as well as the non-linearity of the log transform, make the priors difficult to interpret just by looking at them. Prior predictive simulation is the easiest way to see how the priors interact by seeing what sort of data they generate.

Figure 11 shows the observed entropy data y alongside ten datasets y_{rep} generated based on these priors for α and σ (as well as the prior for β that will be determined below). Clearly the prior predictive distributions are much more spread-out, much more strongly positive, than the observed data. However, more liberal priors are better than priors that hamper the generative process by assigning (near-)zero probabilities to values that we do not actually wish to disqualify (Jackman 2009: 18).[9]

[9] This rule of thumb is known as Cromwell's Rule, named by Lindley (1985) after Oliver Cromwell, an English general who led a campaign against the Scottish army in 1650 and wrote in a letter to the Church of Scotland, "I beseech you, in the bowels of Christ, consider it possible that you are mistaken" (Jackman 2009: 18). Wider priors afford us this possibility.

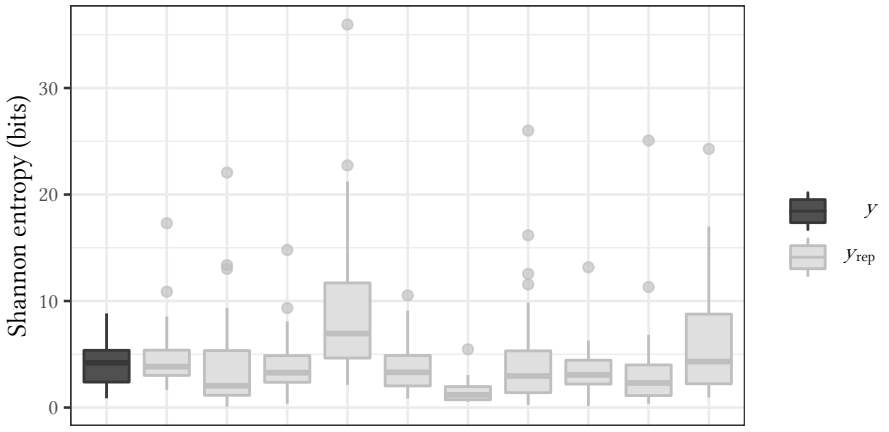

Figure 11: Prior predictive distributions for entropy

The prior for the three β coefficients must be selected with more care. Because they estimate how each predictor affects entropy, they are the main parameters we wish to interpret. But the posterior estimations, and thus the interpretation, depend greatly on the priors we choose.

It is often advisable to use knowledge elicited from experts or gained from previous studies to set priors (Spiegelhalter et al. 2004: Chapter 5, Nicenboim et al. 2021: Chapter 6), but since this is the first study to test the effect of these factors on entropy, I did not know *a priori* how large the effects may be. To find priors that are unspecific enough that they do not unduly bias the βs' posterior estimates, with the goal to regularise but let the data speak for itself, I conducted a sensitivity analysis (Spiegelhalter et al. 2004: Section 5.6, McElreath 2020: 35).

A sensitivity analysis considers the effect on posterior estimates of a range of priors. Figure 12 shows, for five different priors, the posterior estimates of each β. (Since I standardised the predictors to z-scores, I could use the same prior for all three βs.) All of the priors were normally distributed, centered at zero, with standard deviations ranging from 0.01 to 1. Based on this sensitivity analysis, I selected the prior that was most informative without being over-informative, the one for which the posterior has just begun to stabilise, namely $Normal(0, 0.5)$.

Bringing all of these components together, Equation 6 shows the final model.

$$\begin{aligned} entropy_n &\sim LogNormal(\alpha + (z\_fr_n \cdot \beta_{fr}) + (z\_sm_n \cdot \beta_{sm}) + (z\_jp_n \cdot \beta_{jp}), \sigma) \\ \alpha &\sim Normal(1, 0.5) \\ \sigma &\sim Normal_+(0, 0.3) \\ \beta &\sim Normal(0, 0.5) \end{aligned} \tag{6}$$

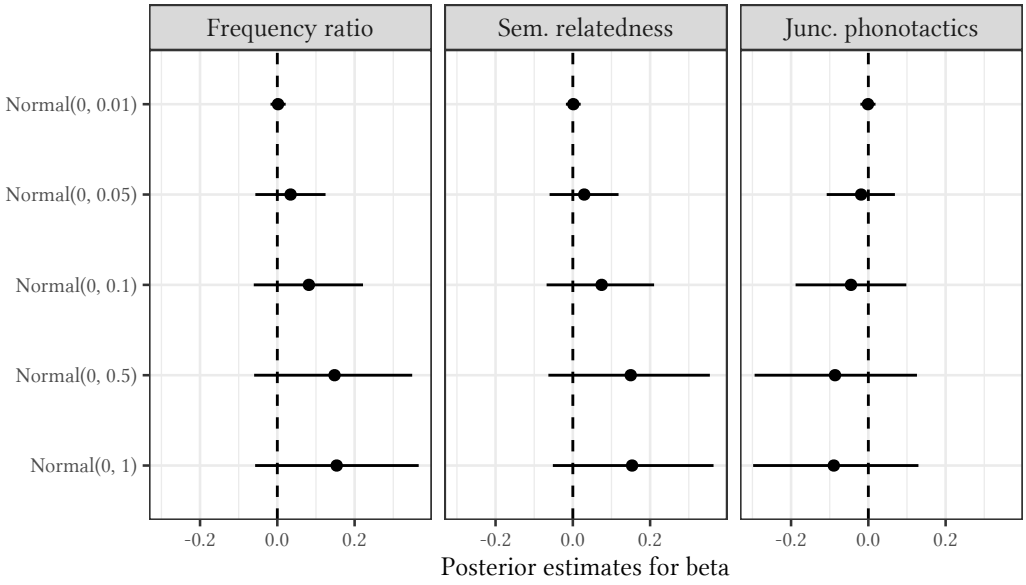

Figure 12: Sensitivity analysis of the β coefficients for differently informative priors; points indicate the posterior mean and error bars give the 95% credible intervals

3.2.3 Effect size estimation

I fit this model in R (R Core Team 2019) via the `brms` interface (Bürkner 2017, 2018) to `Rstan` (Stan Development Team 2020). Since the model is very simple, 3,000 iterations (1,500 of those as warm-up) were enough for the chains to converge; values of the convergence statistic $\hat{R}$ all equalled 1 (McElreath 2020: 281).

Figure 13 visualises the densities of the posterior estimates for the three β parameters in the model's log space. All of the 95% credible intervals contain zero, meaning that we cannot be certain that the effect goes in one direction or the other. However, the model places the majority of the posterior probability mass above zero for the effects of frequency ratio and semantic relatedness, and below zero for the effect of junctural phonotactics. These tendencies are in line with the hypotheses laid out in Table 1.

Table 2 shows the estimated effects on the original scale: what is the change in entropy in bits with a change of one standard deviation in each predictor? Because of the non-linear transformation, the effect on entropy would be different if it were evaluated at different parts of the predictors' range. The effect shown is the predicted difference in entropy between each predictor at its mean and at one standard deviation below the mean, holding the other parameters at their means of zero.

However, merely estimating the direction or size of an effect is not the same as gathering evidence for that effect. To quantify how much evidence we have for or against the effect of each of these predictors, we turn in the next section to Bayes factors.

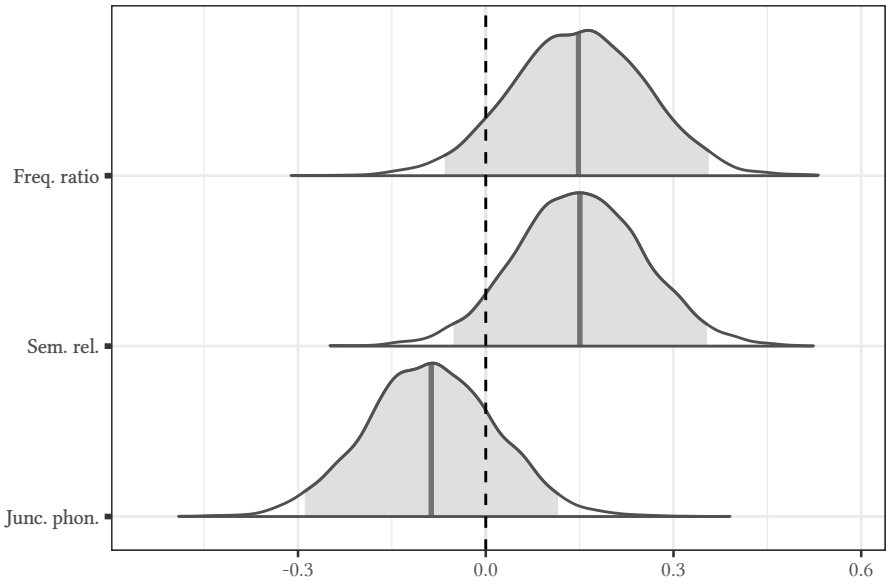

Figure 13: Posterior densities of each predictor's effect on entropy; vertical bars indicate posterior means, and the shaded areas are the 95% credible intervals

Table 2: Posterior estimates of each predictor's effect on entropy, backtransformed to the original scale (i.e., the predicted change in bits when moving from one standard deviation below each predictor's mean to the mean)

Factor	Mean	2.5%	97.5%
Frequency ratio	0.45	−0.22	1.04
Sem. relatedness	0.45	−0.17	1.02
Junc. phonotactics	−0.33	−1.15	0.37

3.2.4 Evidence for effects using Bayes factors

A Bayes factor is a measure of relative likelihood of two models, each of which encodes a specific hypotheses about how the data was generated (Schad et al. 2021, Spiegelhalter et al. 2004: 55). It allows us to answer the question: assuming both models are equally likely *a priori*, how much more likely is it that a model $\mathscr{M}_1$ generated the observed data compared to another model $\mathscr{M}_2$?

The Bayes factor is computed as

$$BF_{12} = \frac{p(y|\mathscr{M}_1)}{p(y|\mathscr{M}_2)},$$

where $p(y|\mathscr{M}_1)$ and $p(y|\mathscr{M}_2)$ are the marginal likelihoods of models $\mathscr{M}_1$ and $\mathscr{M}_2$: the likelihood that each model generated the observed data y, averaging over all possible parameter values (or, in other words, marginalising those values out, hence the name; Schad et al. 2021: 20). The result of this ratio is a continuous measure of evidence, asking "how much evidence do we have for this effect?", rather than a binary decision of "is there evidence or isn't there?" (although binary decisions could also be made, e.g., by imposing some threshold on the value of the Bayes factor or, a more sophisticated approach, by employing utility functions; see Schad et al. 2021).

I conducted three Bayes factor analyses, one for each predictor. In each case, I compared a model that contains all three predictors, representing the alternative hypothesis H1, with a model that contains only the other two predictors, assuming no effect of the predictor we wish to test, representing the null hypothesis, H0.

Since the marginal likelihood tells us how likely a given model is to have generated the observed data, it is very sensitive to the priors encoded in that model. Priors that are narrower or broader than the observed effects will have a smaller marginal likelihood than priors with magnitudes that match the observed effect. Consequently, one must be very cognizant of the priors one chooses when computing and interpreting Bayes factors.

As mentioned above, I had no *a priori* expectations about the size of the effects. So, rather than predefining one set of priors, I performed a prior sensitivity analysis for the Bayes factor as well, computing Bayes factors for a range of different priors for β (Nicenboim et al. 2021: Chapter 16.2). As before, the priors were all normal distributions centered at zero, and their standard deviations ranged from 0.01 to 1.

Figure 14 displays the Bayes factors for each predictor as a function of the width of the priors for β. A value greater than 1 indicates evidence for H1—the alternative model being more likely to have generated the data—while a value less than 1 indicates evidence for H0—the null model being more likely to have generated the data. For the predictors frequency ratio and semantic relatedness, when the priors have about the

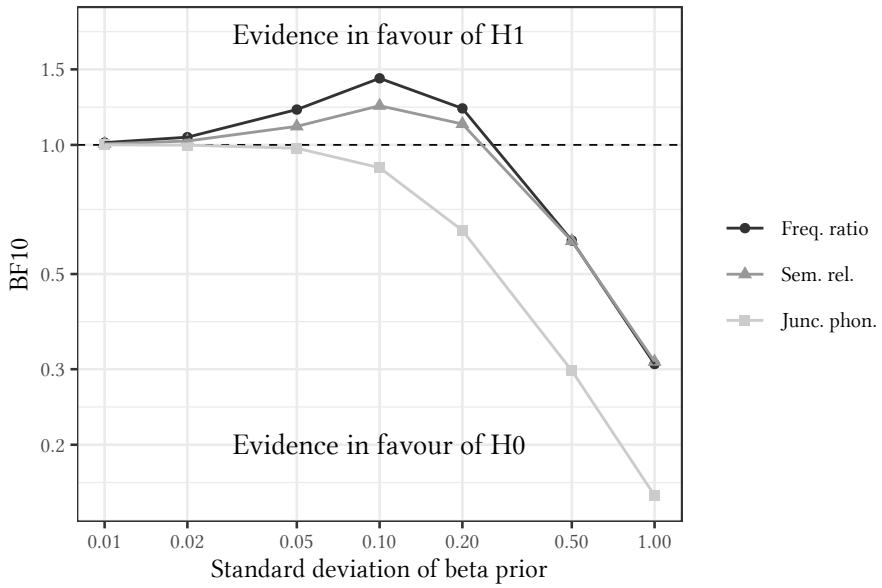

Figure 14: Prior sensitivity analyses for the Bayes factor

same magnitude as the observed effects, the Bayes factors are greater than one, but by so little that the evidence is "not worth more than a bare mention", in the words of Jeffreys (1961: 432). And otherwise, the evidence in favour of H0 grows. (For reference, a common decision threshold for concluding that there is evidence for an effect is a Bayes factor of 10 or 1/10; Schad et al. 2021: 7). For junctural phonotactics, there is no evidence at all for H1, and increasingly substantial evidence for H0.

Thus, all in all, the Bayes factors do not offer evidence in support of the three productivity factors in predicting entropy. In all cases, the null model—the model without each effect—was either nearly as likely or moderately more likely to have generated the data we observed.

3.2.5 Conclusions

With the uncertainty in the effect estimation and the lack of evidence for each predictor from the Bayes factors, this analysis does not provide a resounding validation of entropy as a productivity measure. Nevertheless, I believe that the tendencies in the expected directions still give cause for cautious optimism.

It is also worth noting that the uncertain outcome is not surprising, given the limited number of observations: only 33. This number was kept small to keep the scope of the project manageable, but in principle, many more derivational morphemes could be used—DErivBase lists 234 for German (Zeller et al. 2013). The more data we have, the more we can learn, so including additional morphemes would allow a more conclusive investigation of these three productivity factors and their effect on entropy.

4 Applicability: Using entropy in practice

In the last two sections, we have seen that as sample size increases, entropy approaches a stable, asymptotic value, and that entropy tends toward being affected in the expected ways by factors that we know to affect productivity. I have called these two properties interpretability and validity, respectively. We now turn to the third and final property that makes entropy a good measure of productivity: applicability. In this section, I will show how to identify whether entropy has already stabilised for any given sample and discuss what we can learn even if it has not. These points will be illustrated in two case studies: one in Section 4.1 using synchronic data from DECOW16B (the same data that was used in the Bayesian model above), and one in Section 4.2 using diachronic data from the Early New High German corpus RIDGES (Odebrecht et al. 2017). Example code for running the recommended analyses in both Python and R is given in Appendix E.

4.1 Comparing productivity synchronically

Many productivity studies ask the question: at some fixed point in time, is one morpheme more productive than another? To answer this question, we likely have to compare data from samples of different sizes. For example, the sample sizes of the 35 German suffixes analysed above range from 37 to 19,835. In order to compare the entropy values we obtain for these suffixes, we must be sure that, in all samples, entropy has already stabilised. To determine this, I will propose that we can once again use bootstrapping.

To see whether the entropy obtained for a sample s containing n tokens has begun to stabilise, resample $n/2$ tokens from s some number of times (here I use 100). Compute the entropy of the type frequency distribution of each of the bootstrapped samples, and compare the range of these 100 values to the entropy originally obtained for s. If the entropy of s's type frequency distribution is contained within that range, then we can be confident that s is large enough to provide a stable estimate. However, if the entropy of s's type frequency distribution is greater than that of the bootstrapped samples, we know that the sampled distribution has yet to stabilise.

I conducted this diagnostic for the 35 suffixes in my data. All of their entropy values at sample size n are contained in the range obtained for 100 resamples of size $n/2$. This tells us that the entropy values for all suffixes can reasonably be compared, despite the dramatic differences in sample size. Thus, statements such as the following are valid: "Based on data in DECOW16B, *-ung* has an entropy of 8.84 bits, making it slightly more productive in that corpus than *-er*, which has an entropy of 7.84 bits, and a great deal more productive than *-end* with its entropy of 1.57 bits." (See Appendix D for a listing of the entropy of all suffixes.)

This situation is straightforward; all samples are large enough that the entropy values of the type frequency distributions have stabilised. However, in historical corpus linguistics, large samples are a luxury that is generally not afforded. So, I turn next to a case study with diachronic data, and with it I will illustrate what we can still learn using entropy, even if sample sizes are small.

4.2 Tracking productivity diachronically

Another common scenario in the productivity literature is the question of how the productivity of a single morpheme changes over time. This line of research involves comparing a sample containing derivations from, for example, the fifteenth century to one from the sixteenth century. Ideally, we could compute the entropy of each sample, and as long as it has stabilised, then we could interpret an increase in entropy as a diachronic growth in productivity and a decrease in entropy as a diachronic loss of productivity. But what if bootstrapping a number of samples of size $n/2$ shows that entropy has not yet stabilised? What can we learn in that situation?

The key insight here is that the values we obtain for entropy monotonically increase with sample size. This was shown above in Figure 3. Because of this monotonic increase, we know that any value that we obtain for entropy represents a lower bound: the ultimate entropy for a given morpheme might be more than what we observe at small sample sizes, but it will not be less. This knowledge allows us, in certain situations, to still draw tentative conclusions about productivity, even when entropy has not yet stabilised.

In which situations? When the entropy values over time show an upward trend. To get a visual intuition, consider Figure 15; it illustrates the types of linear trends in productivity that are compatible with two hypothetical diachronic datasets. Downward trends in entropy over time, such as the one schematised in the left-hand plot, are compatible with trend lines in both directions. In contrast, many more positive slopes than negative ones are compatible with entropy that increases over time, as in the right-hand plot, even if the entropy has not yet stabilised for that sample.

Let us now apply this thinking to some historical corpus data. This data comes from Pankratz (2019) and was gathered from the RIDGES corpus (Odebrecht et al. 2017): a collection of Early New High German botanical texts spanning the years 1492–1914. It consists of all tokens in RIDGES of three nominalising morphemes: *-er*, *-heit*, and *-ung*. How the productivity of these morphemes changed in this period of German is interesting because the data contained in RIDGES represents the crystallisation of a new, scientific register of German. And, since scientific language relies heavily on nominalisations, it is possible that these three morphemes undergo a change in productivity during this time. What can we learn about this possible change?

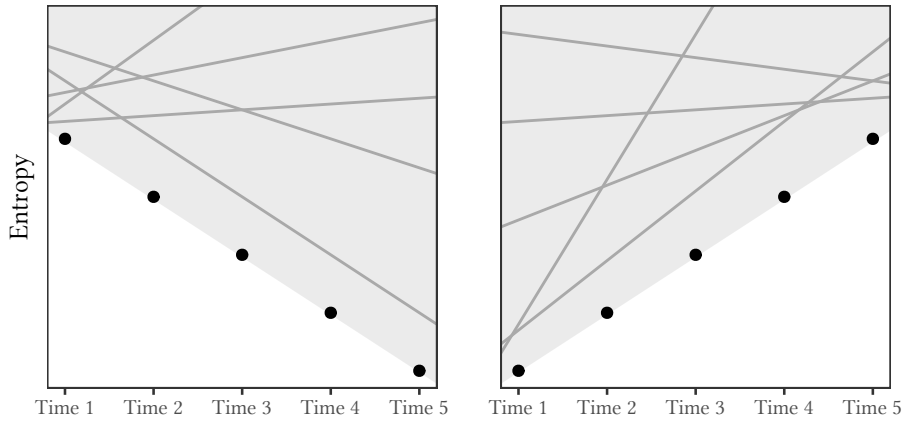

Figure 15: Linear trends that are compatible with changing values of entropy over time; the points represent observed but not yet stabilised values of entropy, the shaded regions represent the possible range of values that entropy might stabilise to, and the grey lines are possible trends within those regions

I binned the data into 100-year samples and calculated the entropy for each sample. Additionally, I bootstrapped these samples, dividing them by factors of two and computing the entropy of each of those smaller samples. The result is shown in Figure 16. For *-er* and *-heit*, entropy values tend to decrease over time. However, the ultimate value of entropy could fall anywhere above those points, so we cannot learn much from this data about how the productivity of those two morphemes has changed. In contrast, for *-ung*, we see an upward trend over time. And because these values are lower bounds, we can tentatively say that entropy probably does not decrease diachronically. Thus, at least for the sort of data in RIDGES, *-ung* may have become more productive over time.

Although there is nothing specifically Bayesian about this look at diachronic data, the analysis still benefits from the Bayesian attitude of being comfortable with uncertainty. The method I advocate here allows researchers to identify how much they can trust the entropy value they obtain—a valuable piece of information, especially when dealing with the small samples that are characteristic of historical corpus studies.

5 Outlook

5.1 Summary

In this thesis, I proposed that the Shannon entropy of a morpheme's type frequency distribution is a suitable way to measure that morpheme's productivity. I motivated this proposal by outlining three characteristics that entropy has: interpretability, validity, and applicability.

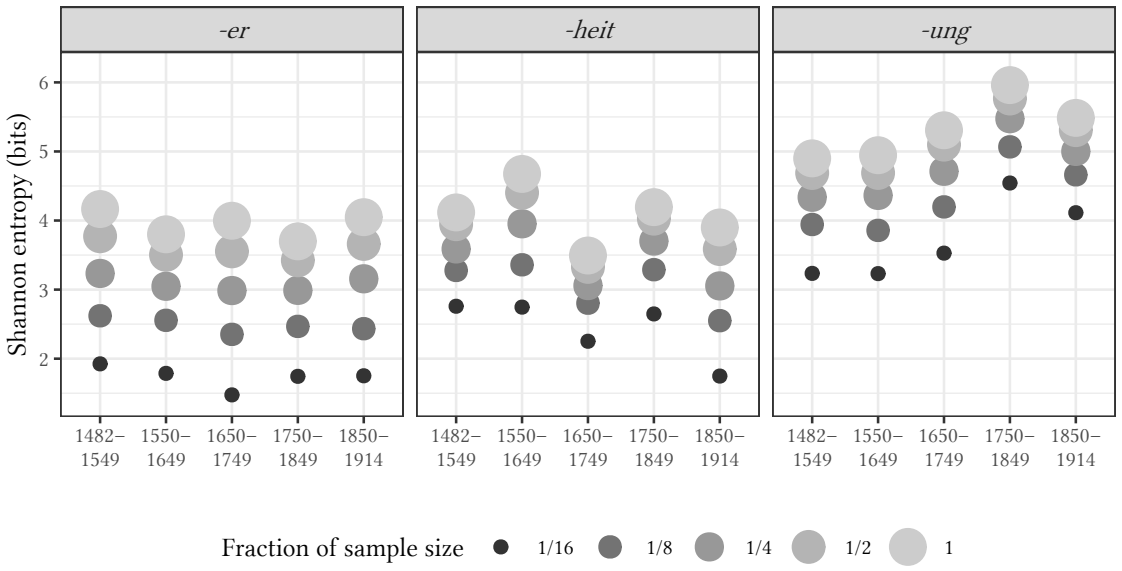

Figure 16: Mean entropy values of small Early New High German samples from RIDGES

First, entropy is interpretable because the shape of a type frequency distribution—what entropy summarises—is closely linked to the productivity of the morpheme that produced that distribution. A more productive morpheme creates more hapax legomena, and thus a more spread-out distribution with a longer tail, leading to to greater entropy. A less productive morpheme, on the other hand, has fewer hapax legomena and most of its tokens are concentrated at the higher frequency ranks, leading to lower entropy. Another important aspect of entropy's interpretability is the main point of this thesis: once the sample size is large enough, entropy is sample-size-independent. It stabilises at a value that reflects a morpheme's productivity.

Second, entropy is tentatively conceptually valid, since it is associated—albeit loosely—with three factors that affect productivity. A Bayesian linear regression analysis showed that the frequency ratio between base and derivation, the semantic relatedness of base and derivation, and the phonotactic properties of the juncture between stem and affix all show tendencies toward predicting entropy in the expected directions.

Third and finally, entropy is applicable to both synchronic and diachronic data, comparing morpheme to morpheme or century to century. And even when sample sizes are too small for entropy to have fully stabilised, by recognising that whatever value is obtained is a lower bound, we can still draw some limited conclusions.

In sum, I believe that entropy is a valuable addition to the arsenal of tools available to researchers of morphological productivity. Before concluding, I outline two avenues that this thesis has opened for future research.

5.2 Directions for future work

5.2.1 Data from German and other languages

The scope of the present study was limited substantially by the amount of work required to identify the bases for each derivation in the samples from DECOW16B. Since no tools existed to do this automatically, I created every backformation rule from scratch. If a tool were to be developed for German that could do this automatically, then gathering data for more suffixes than the 35 studied here would become trivial.

Further, it would be interesting to expand the scope of the study to word formation in other languages, should the necessary datasets exist. Do the three factors I tested here influence productivity in the same way in all languages? Is word formation on the whole equally productive in different languages? With entropy in our toolkit, we may now be equipped to quantitatively approach such questions.

5.2.2 Predicting when entropy will stabilise

I showed in Section 4 that we can estimate whether entropy has stabilised for a given sample by bootstrapping samples of size $n/2$ and checking whether the entropy value obtained for the full sample is contained within the range of values obtained for a sample half the size. But is there also a mathematical law that we can use to identify the sample size at which entropy stabilises?

One simple function that approximates entropy y as a function of sample size x (inspired by Wickelgren 1977: Figure 1) is

$$y(x) = \lambda(1 - e^{-\beta \log(x)}), \tag{7}$$

where $\lambda > 0$ is the asymptote that the curve approaches and $\beta > 0$ is the rate of approach. Figure 17 shows this curve overlaid with the mean entropy estimates from the bootstrapping procedure for *-heit*, *-schaft*, and *-nis* described in Section 2.2. (The parameters λ and β were determined by trial and error.) As these plots show, this approximation is inexact: entropy is overestimated at small sample sizes. Nevertheless, I will use it as an illustration.

Our aim is to estimate the sample size at which entropy begins to stabilise. To do this, we can take the first derivative of the curve to identify the rate at which entropy is changing, and then identify the value of x, that is, the sample size, at which the derivative falls below some near-zero threshold.

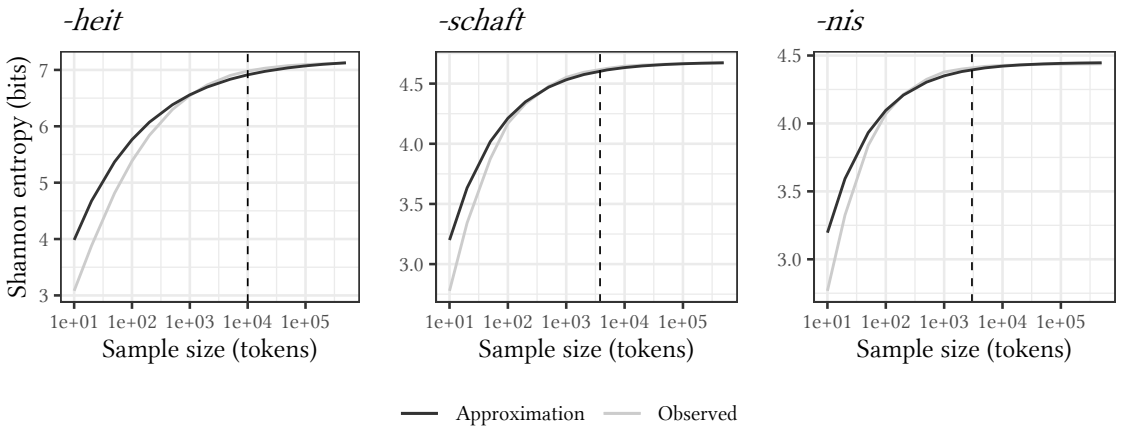

Figure 17: Approximated vs. observed entropy as a function of sample size; dashed lines indicate the estimated stable sample size

Table 3: Estimates of sample size needed for entropy to stabilise

Suffix	λ	β	Sample size
-heit	7.20	0.35	10,024
-nis	4.45	0.55	2,996
-schaft	4.68	0.50	3,797

The first derivative of Equation 7 is given in Equation 8 (see Appendix F for the working-out).

$$\frac{dy}{dx} = \frac{\lambda}{x}\left(\beta e^{-\beta \log(x)}\right) \tag{8}$$

Setting this expression equal to c, the desired near-zero slope of a tangent line to the exponential function, and solving for x (again, see Appendix F) yields

$$x = \left(\frac{\lambda\beta}{c}\right)^{\frac{1}{\beta+1}}. \tag{9}$$

Whatever near-zero threshold we select for c has to be very near zero indeed, since the magnitude of the x axis scale is very large. With $c = 1 \times 10^{-5}$ and the pre-determined values of λ and β for each of the three suffixes shown in Figure 17, Equation 9 evaluates as shown in Table 3. The estimated sample sizes, superimposed as vertical lines on the curves in Figure 17, look like acceptable approximations of when the entropy begins to stabilise.

If we already have samples for which entropy has stabilised, then either the bootstrapping method from Section 4 or this mathematical method can be used. However, it would be particularly useful to be able to extrapolate such a curve from a small sample, and then use that curve to estimate how many more tokens would likely be needed for a sufficiently large sample. To do this, we would need to find a function that more accurately approximates entropy as a function of sample size, and we would need a way to extrapolate that curve from a limited sample (perhaps with a method along the lines of Evert 2004). These two steps would be promising paths for future work.

5.3 Conclusion

Part of language users' linguistic competence is their knowledge of morphological productivity: how readily a morpheme can be used to form new words. This knowledge is reflected in language usage and can thus be quantified based on usage data from corpora (Baayen 1989). I have proposed a novel way of measuring morphological productivity that eventually does not depend on the sample size used to compute it—a clear improvement over the productivity measures that are the current standard.

Nevertheless, as many researchers have already observed, productivity is a multifaceted phenomenon (Baayen 2001, Bauer 2001, Zeldes 2012), and as Kempf's flowchart in Figure 6 showed, many aspects of a word formation process' behaviour must be considered in a full understanding of productivity—many more than those that can be quantified using corpora. What I have provided here is a solution to one single problem, one single aspect of the complex that is morphological productivity. There is still a great deal left to understand.

References

Adamic, Lada A. 2000. *Zipf, Power-Laws, and Pareto – A Ranking Tutorial.* Technical report. Palo Alto, CA: Information Dynamics Lab, HP Labs.

Baayen, R. Harald. 1989. *A corpus-based approach to morphological productivity: Statistical analysis and psycholinguistic interpertation.* Vrije Universiteit Amsterdam PhD thesis.

Baayen, R. Harald. 2001. *Word frequency distributions.* Dordrecht: Kluwer Academic Publishers.

Baayen, R. Harald. 2009. Corpus linguistics in morphology: Morphological productivity. In Anke Lüdeling & Merja Kytö (eds.), *Corpus linguistics: An international handbook*, 900–919. Berlin: De Gruyter.

Baayen, R. Harald & Robert Schreuder. 2000. Towards a Psycholinguistic Computational Model for Morphological Parsing. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences* 358(1769). 1281–1293.

Baroni, Marco. 2009. Distributions in text. In Anke Lüdeling & Merja Kytö (eds.), *Corpus linguistics: An international handbook*, 803–822. Berlin: De Gruyter.

Baroni, Marco & Stefan Evert. 2016. *Unit 5: Word Frequency Distributions with the zipfR Package (Statistics for Linguists with R – a SIGIL Course).* `https://www.stefan-evert.de/SIGIL/sigil_R/materials/05_zipfr.slides.pdf`.

Bauer, Laurie. 2001. *Morphological Productivity.* 1st edn. Cambridge: Cambridge University Press.

Biemann, Chris & Uwe Quasthoff. 2009. Networks generated from natural language text. In *Modeling and Simulation in Science, Engineering and Technology*, 167–185.

Bonachela, Juan A, Haye Hinrichsen & Miguel A Muñoz. 2008. Entropy estimates of small data sets. *Journal of Physics A: Mathematical and Theoretical* 41. 202001.

Bürkner, Paul-Christian. 2017. brms: An R package for Bayesian multilevel models using Stan. *Journal of Statistical Software* 80(1). 1–28.

Bürkner, Paul-Christian. 2018. Advanced Bayesian multilevel modeling with the R package brms. *The R Journal* 10(1). 395–411.

Bybee, Joan & Clay Beckner. 2009. Usage-based theory. In Berndt Heine & Heiko Narrog (eds.), *The Oxford Handbook of Linguistic Analysis*, 827–855. Oxford: Oxford University Press.

Cowie, Claire. 1999. *Diachronic word-formation: A corpus-based study of derived nominalizations in the history of English.* University of Cambridge PhD thesis.

Cowie, Claire & Christiane Dalton-Puffer. 2002. Diachronic word-formation and studying changes in productivity over time: Theoretical and methodological considerations. In Javier E. Díaz Vera (ed.), *A changing world of words: Studies in English historical lexicography, lexicology and semantics*, 410–437. Amsterdam: Rodopi.

Dal, Georgette & Fiammetta Namer. 2016. Productivity. In Andrew Hippisley & Gregory T. Stump (eds.), *The Cambridge Handbook of Morphology* (Cambridge Handbooks in Language and Linguistics), 70–89. Cambridge: Cambridge University Press.

Demske, Ulrike. 2001. *Merkmale und Relationen: Diachrone Studien zur Nominalphrase des Deutschen*. Berlin: De Gruyter.

Diessel, Holger. 2017. Usage-Based Linguistics. In *Oxford Research Encyclopedia of Linguistics*, 1–29. Oxford: Oxford University Press.

Doerfert, Regina. 1994. *Die Substantivableitung mit -heit, -keit, -ida, -î im Frühneuhochdeutschen*. Berlin: De Gruyter.

Efron, Bradley. 1979. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics* 7(1). 1–26.

Evert, Stefan. 2004. A simple LNRE model for random character sequences. In *Proceedings of JADT 2004: 7es Journées internationales d'Analyse statistique des Données Textuelles*, 1–12.

Evert, Stefan & Marco Baroni. 2020. *Package 'zipfR': Statistical Models for Word Frequency Distributions*.

Fleischer, Wolfgang & Irmhild Barz. 2012. *Wortbildung der deutschen Gegenwartssprache*. 4th edn. (De Gruyter Studium). Berlin: De Gruyter.

Gaeta, Livio & Davide Ricca. 2006. Productivity in Italian word formation: A variable-corpus approach. *Linguistics* 44(1). 57–89.

Gelman, Andrew & Jennifer Hill. 2007. *Data analysis using regression and multilevel/hierarchical models*. Cambridge: Cambridge University Press.

Goodfellow, Ian, Yoshua Bengio & Aaron Courville. 2016. *Deep learning* (Adaptive Computation and Machine Learning). Cambridge, Mass.: The MIT Press.

Hartmann, Stefan. 2016. *Wortbildungswandel: Eine diachronie Studie zu deutschen Nominalisierungsmustern* (Studia Linguistica Germanica 125). Berlin: De Gruyter.

Hay, Jennifer. 2001. Lexical frequency in morphology: Is everything relative? *Linguistics* 39(6).

Hay, Jennifer & Harald Baayen. 2002. Parsing and Productivity. In Geert Booij & Jaap Van Marle (eds.), *Yearbook of Morphology 2001*, 203–235. Dordrecht: Kluwer Academic Publishers.

Hay, Jennifer & Harald Baayen. 2003. Phonotactics, parsing and productivity. *Italian Journal of Linguistics* 1. 99–130.

Hein, Katrin & Annelen Brunner. 2020. Why do some lexemes combine more frequently than others? – An empirical approach to productivity in German compound formation. In *Proceedings of Mediterranean Morphology Meetings 12*, 28–41.

Jackman, Simon. 2009. *Bayesian analysis for the social sciences*. John Wiley & Sons.

Jeffreys, Harold. 1961. *Theory of probability*. 3rd edn. (Oxford Classic Texts in the Physical Sciences). Oxford: Oxford University Press.

Jordan, D. W. & P. Smith. 2008. *Mathematical techniques*. 4th edn. Oxford: Oxford University Press.

Kempf, Luise. 2016. *Adjektivsuffixe in Konkurrenz: Wortbildungswandel vom Frühneuhochdeutschen zum Neuhochdeutschen*. Vol. 126 (Studia Linguistica Germanica). Berlin: de Gruyter.

Lindley, Dennis V. 1985. *Making decisions*. 2nd edn. London: John Wiley & Sons, Ltd.

Lüdeling, Anke. 2009. Carmen Scherer, Wortbildungswandel und Produktivität. *Beiträge zur Geschichte der deutschen Sprache und Literatur (PBB)* 131(2). 333–339.

Mandelbrot, Benoit. 1953. An informational theory of the statistical structure of language. *Communication Theory* 84. 486–502.

McElreath, Richard. 2020. *Statistical rethinking: A Bayesian course with examples in R and Stan*. 2nd edn. (CRC Texts in Statistical Science). Boca Raton: CRC Press.

Milin, Petar, Dagmar Divjak, Strahinja Dimitrijević & R. Harald Baayen. 2016. Towards cognitively plausible data science in language research. *Cognitive Linguistics* 27(4). 507–526.

Mitchell, Melanie. 2009. *Complexity: A guided tour*. Oxford: Oxford University Press.

Nicenboim, Bruno, Daniel J. Schad & Shravan Vasishth. 2021. *An introduction to Bayesian data analysis for cognitive science*. Online draft of 2021-06-09.

Odebrecht, Carolin, Malte Belz, Amir Zeldes, Anke Lüdeling & Thomas Krause. 2017. RIDGES Herbology: Designing a diachronic multi-layer corpus. *Language Resources and Evaluation* 51(3). 695–725.

Pankratz, Elizabeth. 2019. *A study of diachronic changes in the productivity of several Early New High German derivational morphemes using the RIDGES corpus*. Humboldt-Universität zu Berlin Master's thesis.

Plag, Ingo. 1999. *Morphological productivity: Structural constraints in English derivation*. Berlin: De Gruyter.

Plag, Ingo, Christiane Dalton-Puffer & Harald Baayen. 1999. Morphological productivity across speech and writing. *English Language and Linguistics* 3(2). 209–228.

R Core Team. 2019. *R: A Language and Environment for Statistical Computing*. Vienna, Austria.

Riehemann, Susanne Z. 1998. Type-based derivational morphology. *Journal of Comparative Germanic Linguistics* 2(1). 49–77.

Roulston, Mark S. 1999. Estimating the errors on measured entropy and mutual information. *Physica D* 125. 285–294.

Saffran, Jenny R., Richard N. Aslin & Elissa L. Newport. 1996a. Statistical learning by 8-month-old infants. *Science* 274(5294). 1926–1928.

Saffran, Jenny R., Elissa L. Newport & Richard N. Aslin. 1996b. Word segmentation: The role of distributional cues. *Journal of Memory and Language* 35(4). 606–621.

Schad, Daniel J., Bruno Nicenboim, Paul-Christian Bürkner, Michael Betancourt & Shravan Vasishth. 2021. *Workflow Techniques for the Robust Use of Bayes Factors*. `http://arxiv.org/abs/2103.08744` (3 July, 2021).

Schäfer, Roland. 2015. Processing and querying large web corpora with the COW14 architecture. In Piotr Bański, Hanno Biber, Evelyn Breiteneder, Marc Kupietz, Harald Lüngen & Andreas Witt (eds.), *Proceedings of Challenges in the Management of Large Corpora 3 (CMLC-3)*, 28–34.

Schäfer, Roland & Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, 486–493. Istanbul, Turkey: European Language Resources Association (ELRA).

Scherer, Carmen. 2005. *Wortbildungswandel und Produktivität: Eine empirische Studie zur nominalen* -er*-Derivation im Deutschen*. Tübingen: Niemeyer.

Scherer, Carmen. 2007. The role of productivity in word-formation change. In Joseph C. Salmons & Shannon Dubenion-Smith (eds.), *Historical Linguistics 2005: Selected papers from the 17th International Conference on Historical Linguistics (Current issues in linguistic theory 284)*. Amsterdam: John Benjamins.

Schmid, Helmut, Arne Fitschen & Ulrich Heid. 2004. SMOR: A German computational morphology covering derivation, composition and inflection. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, 1263–1266.

Schneider-Wiejowski, Karina. 2011. *Produktivität in der deutschen Derivationsmorphologie*. Universität Bielefeld PhD thesis.

Shannon, Claude E. 1948. A mathematical theory of communication. *The Bell System Technical Journal* 27(3). 379–423.

Spiegelhalter, David J., Keith R. Abrams & Jonathan P. Myles. 2004. *Bayesian approaches to clinical trials and health-care evaluation*. Vol. 13. John Wiley & Sons.

Stan Development Team. 2020. *RStan: The R Interface to Stan*.

Tweedie, Fiona J & R Harald Baayen. 1998. How variable may a constant be? Measures of lexical richness in perspective. *Computers and the Humanities* 32(5). 323–352.

Wickelgren, Wayne A. 1977. Speed-accuracy tradeoff and information processing dynamics. *Acta psychologica* 41(1). 67–85.

Würzner, Kay-Michael & Bryan Jurish. 2015. A hybrid approach to grapheme-phoneme conversion. In *Proceedings of the 12th International Conference on Finite State Methods and Natural Language Processing*.

Zeldes, Amir. 2012. *Productivity in argument selection: From morphology to syntax* (Trends in Linguistics. Studies and Monographs 260). Berlin: De Gruyter.

Zeller, Britta, Sebastian Padó & Jan Šnajder. 2014. Towards Semantic Validation of a Derivational Lexicon. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 1728–1739.

Zeller, Britta, Jan Šnajder & Sebastian Padó. 2013. DErivBase: Inducing and Evaluating a Derivational Morphology Resource for German. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1201–1211.

Zipf, George Kingsley. 1949. *Human behavior and the principle of least effort*. Cambridge, Mass.: Addison-Wesley Press, Inc.

Appendices

A Bootstrapping with and without replacement

Bootstrapping offers a way of estimating the sampling distribution of a measure such as entropy using random resampling, conventionally done with replacement (Efron 1979). However, Baroni & Evert (2016: 94) state that sampling with replacement is not appropriate for type-token distributions because it underestimates the total vocabulary size (i.e., number of types). This may be because, when we take the population to be all of natural language, the samples will always contain new types; the language's full vocabulary will not be exhausted, even by very large corpora (Baroni 2009: 818). In contrast, when we resample from some finite sample, we take that sample to be the population, and there, it is very possible to exhaust the limited vocabulary within it. No new types can be included that were not in the original sample.

Sampling without replacement is more akin to the first situation, since it forces us to explore more of the given sample, making us more likely to find any as-yet-unseen types compared to when we can visit already-seen tokens again. Thus, sampling without replacement leads to a greater diversity in types than does sampling with replacement.

That said, sampling with or without replacement does not change the pattern of results obtained for the four productivity measures discussed in Section 2.3 (type count, $\mathscr{P}$, S, and entropy). Consider the plots in Figures 18 through 21, in which each of the productivity measures is visualised as a function of sample size and samples generated with and without replacement are overlaid.

Any differences between resampling with and without replacement only become apparent as resample sizes approach the true sample sizes; only then is the potential for replacement realised. And, corroborating Baroni & Evert (2016), it is only the measures that relate directly to the vocabulary—type count and S—that show clear differences at those large resample sizes. In Figure 18, we do indeed see that sampling without replacement leads to fewer types than sampling with replacement. And in Figure 20, we see that S continues its upward trajectory when samples are drawn without replacement. This is presumably because in those samples, the type counts are still growing, while in the samples with replacement, fewer new types are being encountered, so the ultimate vocabulary size is estimated to be less.

In both cases, though, the conclusions from Section 2.3 still hold. Type count still shows a systematic positive association with sample size when samples are drawn without replacement, as Baroni & Evert (2016) recommend, and, in fact, S shows even more of one.

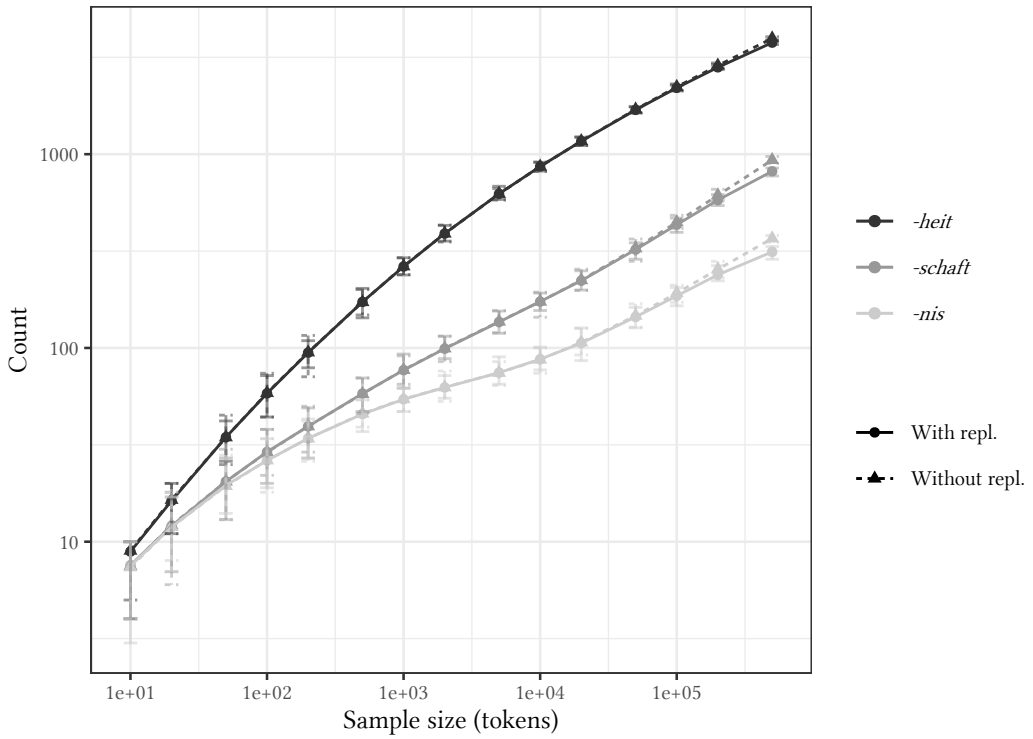

Figure 18: Type count for samples bootstrapped with and without replacement

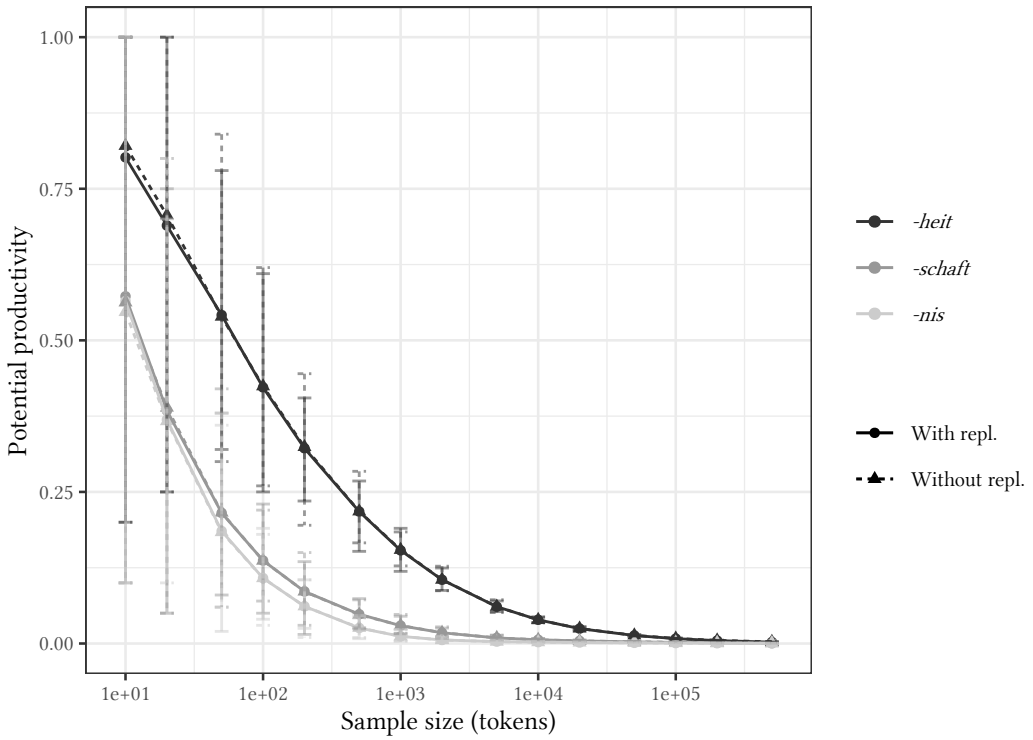

Figure 19: $\mathscr{P}$ for samples bootstrapped with and without replacement

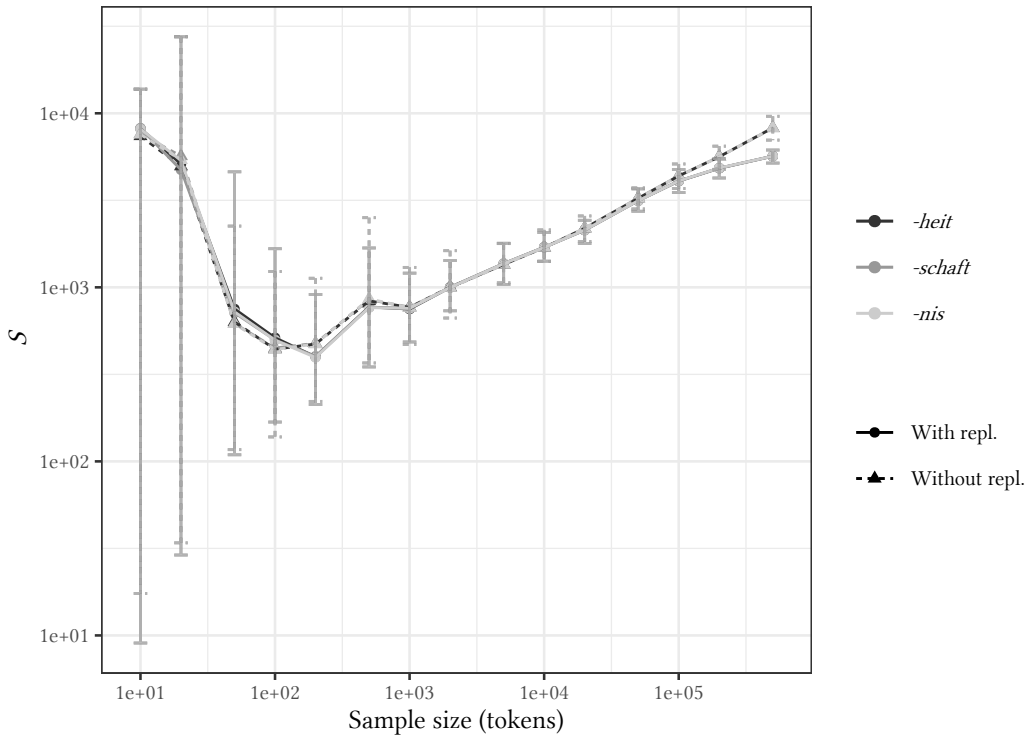

Figure 20: S for samples bootstrapped with and without replacement

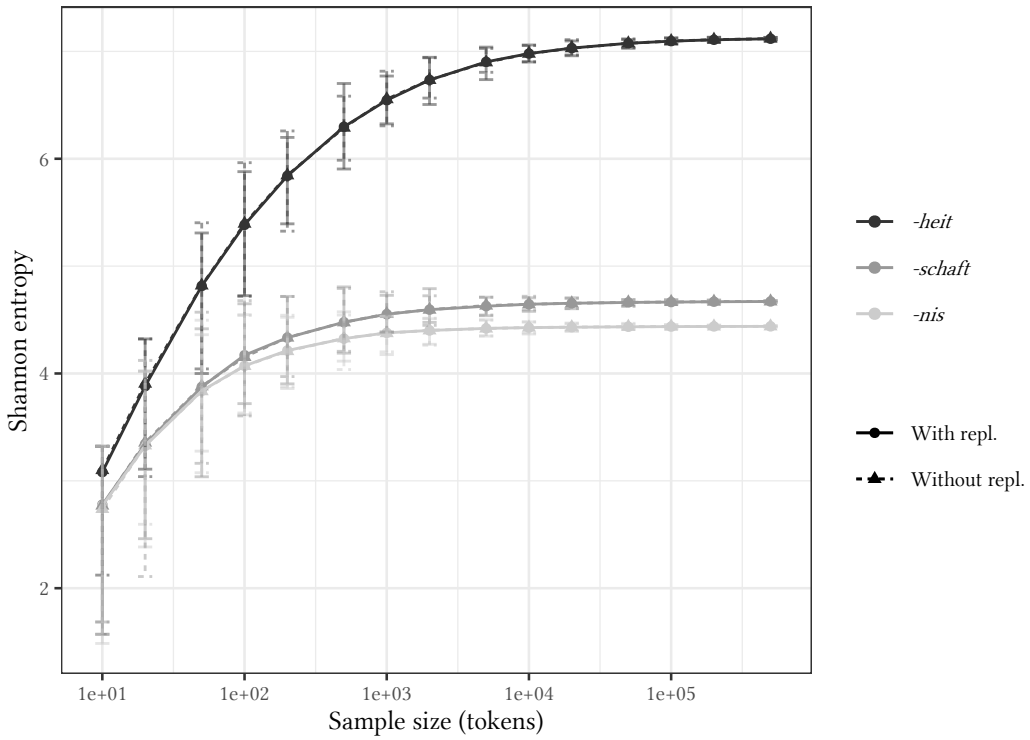

Figure 21: Entropy for samples bootstrapped with and without replacement

B Sampling suffixes from DECOW16B

For the sake of reproducibility, this appendix describes how I created the samples used in this thesis. All code, intermediate and supplementary files, and detailed documentation is available at `https://github.com/epankratz/entropy-productivity-thesis/tree/main/1_data`.

B.1 The three large samples: *-heit*, *-schaft*, and *-nis*

In Sections 2 and 5, I show data for *-heit*, *-schaft*, and *-nis*. Because these samples were used for illustrative purposes rather than for analysis, they were less carefully constructed than were the smaller samples I will discuss in Appendix B.2. I simply drew all of the tokens from DECOW16A-NANO that contain these three suffixes.

I chose these suffixes for two reasons. First, they are orthographically distinctive. I knew that I could not post-process those very large samples very thoroughly, so I aimed to minimise the danger of false positives by choosing suffixes with a more uncommon form than, say, *-e* or *-er*. Second, they also intuitively differ in productivity, so the values returned by productivity measures should vary.

B.2 The 35 smaller samples and `backformer`

Of the over 200 rules for derivational suffixes that DErivBase contains, I selected only nominalisers (those converting either adjectives or verbs into nouns) and then excluded those that were too specific or infrequent to be of interest. This resulted in the selection of the 35 suffixes used in Sections 3 and 4. Note that two of the suffixes are syncretic and are distinguised using subscripts: *-e*$_A$ takes adjectival bases (e.g., *breit* 'wide' ~ *Breite* 'width') while *-e*$_V$ takes verbal bases (e.g., *übernehmen* 'to take over' ~ *Übernahme* 'takeover').

To gather samples for each of these suffixes, I defined queries that would match nouns ending in the desired strings and exclude common non-derivations that also happen to end in those strings (e.g., the query for *-er* excluded, among other things, *September*, *Oktober*, *November*, and *Dezember*). I exported the first 100,000 hits in DECOW16B for each query. Whenever the hit was analysed in the corpus as a nominal compound, I selected only the head noun as the lemma (e.g., *Sprachwissenschaft* 'linguistics' was lemmatised to *Wissenschaft* 'science'; Zeldes 2012: 52).

Despite the manual exclusion of common non-derivations, the samples still contained a substantial amount of noise. I thus used `backformer`, my rule-based Python module for identifying bases of derivations, to distill the samples down to only those words that have a plausible base. The idea was that the existence of a base indicates that the given word really is a legitimate derivation and not a false positive. I defined a plausible base as

a form generated by `backformer` that appeared in DECOW16A-NANO more than zero times and that I then manually annotated as correct. When more than one candidate base was plausible, I selected the one that seemed best, guided by the corpus counts (though see Fleischer & Barz 2012: 69 for discussion of multiple bases motivating a derivation). Whenever `backformer` failed to create an idiosyncratic base (e.g., *Konservatismus* 'conservativism' was backformed to *konservatisch* rather than the correct *konservativ* 'conservative'), that base was manually added to the dataset and its frequency was queried in a follow-up step. (The rules in `backformer` are largely based on those in DErivBase.)

The choice of disqualifying all candidate bases that have a frequency of 0 was not entirely trivial. It is possible that some base is legitimate but so infrequent that DECOW16A-NANO does not contain it, meaning that I would have excluded a valid base-derivation pair from my sample. However, a true base-derivation pair that is genuinely perceived as such would probably not have a base that is so vanishingly rare. And further, this threshold provides a solid criterion to accompany my subjective judgments.

Although an automatic machine-learning approach would have been considerably less time-intensive to apply, the benefit of a rule-based system is that bases can be generated equally well for known and unknown words. In the context of productivity, this ability is particularly important, because many of the words that productive morphemes generate are likely to be unknown.

Returning to the samples: it was prohibitively time-intensive to apply `backformer` to all 100,000 tokens in the original samples, so I randomly selected 20,000 tokens from each. I chose this quantity because, impressionistically, Figure 3 showed that entropy should be stable, even for the more productive suffixes, upward of about 10,000 tokens. (And as it turns out, the largest of the loose mathematical estimates of the stable sample size from Section 5.2.2, the one for *-heit*, is indeed around 10,000.)

Thus, the samples have a maximum size of nearly 20,000 and a minimum size—once all the non-derivations are removed—of 37; see Appendix D.

C Identifying German simplexes

To compute the probability of character bigrams appearing in German simplexes (Section 3.1.3), I needed a list of such simplexes. My general strategy was to start with a list of lemmas and exclude those that are, or are likely to be, multimorphemic. This is a high-precision, low-recall approach: I wanted the lemmas that I got at the end to be simplexes, and it was acceptable if there were some true simplexes that were not included in the final sample.

I began with a list of lemmas in DECOW16B and their frequencies, freely available from the COW website.[10] I filtered this list for only the lemmas with a frequency above 10,000, since impressionistically, there were very few, if any, simplexes with a frequency below that threshold. I then removed any lemmas that began or ended with strings corresponding to German prefixes or suffixes, respectively.

To identify nominal compounds, which should also be excluded, I ran the resulting word list through the morphological analyser SMOR (Schmid et al. 2004). Compounds were removed, along with words that SMOR could not identify, any words that contain numerals or punctuation, and words that have a length of 1. (It was not possible to use SMOR in the first step to also flag lemmas containing prefixes or suffixes, since SMOR is not reliable at identifying derivational morphology; for example, it generally does not separate the adjectival suffix *-lich* from the stem.) Following the SMOR step, I manually annotated the remaining words for simplexhood. The words that were selected in this step were the ones used in the analysis.

It is also important to note that the bigram frequencies were computed per token, rather than per type, since the former is truer to our experience of language and the token-based way that this experience updates our cognitive representations (Bybee & Beckner 2009). To illustrate, take the bigram <tz> in the word *Katze* 'cat'. If *Katze* appeared in the corpus 200 times, then the bigram <tz> would be counted 200 times. A type-wise bigram frequency (in which <tz> is only counted once per simplex it appears in) would overlook informative differences in frequency.

[10] `https://www.webcorpora.org/opendata/frequencies/`

D Data used in the linear model

Suffix	Sample size	Type count	Freq. ratio	Sem. rel.	Junc. phon.	Entropy
-ung	18,927	1436	−0.73	0.94	0.0018	8.84
-er	9,252	751	1.03	0.81	0.0064	7.84
-heit	19,835	1099	2.33	0.97	0.0000	6.94
-e$_V$	5,222	185	−1.66	0.75	0.0080	6.10
-ismus	17,334	361	3.02	0.81	0.0042	6.04
-ie	9,155	300	−0.06	0.97	0.0034	5.91
-ist	15,088	286	2.94	0.79	0.0038	5.61
-ität	10,837	168	2.07	0.94	0.0043	5.50
-ation	15,824	221	−1.22	0.94	0.0019	5.37
-el	7,592	115	−1.28	0.77	0.0063	4.87
-ator	16,585	119	1.73	0.80	0.0020	4.79
-ik	17,999	176	0.95	0.97	0.0045	4.76
-ling	14,833	73	3.36	0.72	0.0006	4.53
-schaft	19,197	156	3.91	0.89	0.0018	4.51
-iker	19,371	147	2.67	0.92	0.0038	4.33
-e$_A$	754	33	1.80	0.91	0.0118	4.24
-nis	19,119	53	0.22	0.80	0.0002	4.20
-ant	8,137	66	1.72	0.80	0.0030	4.04
-eur	3,225	26	0.56	0.84	0.0081	3.43
-ent	11,145	27	−0.76	0.90	0.0080	3.43
-ition	16,758	16	−2.87	0.90	0.0260	2.91
-ateur	13,671	71	4.90	0.86	0.0024	2.91
-enz	7,592	17	−1.98	0.92	0.0072	2.90
-ikum	7,870	62	3.71	0.82	0.0031	2.55
-atur	4,826	16	1.32	0.77	0.0029	2.39
-anz	5,300	16	0.79	0.86	0.0027	2.31
-age	1,403	14	0.90	0.75	0.0011	1.90
-ium	5,171	19	0.49	0.91	0.0446	1.86
-end	88	11	4.82	NA	0.0027	1.57
-and	37	4	−0.81	0.83	0.0017	1.28
-itur	11,240	4	0.93	0.87	0.0064	1.18
-ament	4,452	3	−4.03	0.69	0.0053	1.15
-ement	6,351	7	−0.37	0.88	0.0055	0.97
-iment	12,305	2	−1.17	0.82	0.0009	0.87
-iteur	501	3	4.27	NA	0.0086	0.07

E Example code for an entropy-based analysis

The code below assumes a UTF-8-encoded text file `schaft.txt` that contains a corpus sample with one token per line, e.g.:

```
Wirtschaft
Mitgliedschaft
Genossenschaft
Mannschaft
Wissenschaft
Mannschaft
Mannschaft
Nachbarschaft
Gesellschaft
Herrschaft
⋮
```

E.1 Using Python

```
import numpy as np
import pandas as pd
from scipy.stats import entropy

# Set the number of bootstrapped samples to draw and whether to draw them
# with or without replacement (see Appendix A).
N_RESAMPLE = 100
REPL = True

# A function to convert the given sample to a type frequency distribution
# and to get its Shannon entropy.
def get_entropy(samp):
    freqdist = pd.DataFrame(samp.value_counts())
    return entropy(freqdist, base=2)[0]

# Read in sample and apply get_entropy().
with open('schaft.txt', 'r', encoding='UTF-8') as file:
    full_sample = pd.Series([token.strip() for token in file.readlines()])
full_entropy = get_entropy(full_sample)
```

```
# Bootstrap N_RESAMPLE samples of size RESAMPLE_SIZE, get entropy of each,
# compare with observed min and max values so far and update these if needed.
RESAMPLE_SIZE = int(np.floor(len(full_sample)/2))
boot_min = 1e10
boot_max = 1e-10

for i in range(N_RESAMPLE):
    boot_sample = np.random.choice(full_sample, RESAMPLE_SIZE, replace = REPL)
    boot_entropy = get_entropy(pd.Series(boot_sample))
    boot_min = boot_entropy if boot_entropy < boot_min else boot_min
    boot_max = boot_entropy if boot_entropy > boot_max else boot_max

# Check whether full_entropy is within the range of observed values;
# will print True or False.
print((full_entropy < boot_max) & (full_entropy > boot_min))
```

E.2 Using R

```
library(entropy)
library(readr)

# Set the number of bootstrapped samples to draw and whether to draw them
# with or without replacement (see Appendix A).
N_RESAMPLE <- 100
REPL <- TRUE

# A function to convert the given sample to a type frequency distribution
# and to get its Shannon entropy.
get_entropy <- function(samp){
  freqdist <- as.data.frame(table(samp))
  return(entropy.empirical(freqdist$Freq, unit='log2'))
}

# Read in sample and apply get_entropy().
full_sample <- read_lines('schaft.txt')
full_entropy <- get_entropy(full_sample)
```

```
# Bootstrap N_RESAMPLE samples of size RESAMPLE_SIZE, get entropy of each,
# compare with observed min and max values so far and update these if needed.
RESAMPLE_SIZE <- floor(length(full_sample)/2)
boot_min <- Inf
boot_max <- -Inf

for(i in 1:N_RESAMPLE){
  boot_sample <- sample(full_sample,
                        size = RESAMPLE_SIZE,
                        replace = REPL)
  boot_entropy <- get_entropy(boot_sample)
  boot_min <- ifelse(boot_entropy < boot_min, boot_entropy, boot_min)
  boot_max <- ifelse(boot_entropy > boot_max, boot_entropy, boot_max)
}

# Check whether full_entropy is within the range of observed values;
# will print TRUE or FALSE.
full_entropy < boot_max & full_entropy > boot_min
```

F Mathematical details

I first show the working-out of the first derivative of the function given as Equation 7 above.

$$y(x) = \lambda(1 - e^{-\beta \log(x)})$$

I begin by applying the product rule, which states that if $y(x) = u(x)v(x)$, then $\frac{dy}{dx} = u\frac{dv}{dx} + v\frac{du}{dx}$. Let $u = u(x) = \lambda$ and $v = v(x) = 1 - e^{-\beta \log(x)}$. Conveniently,

$$\frac{du}{dx} = 0,$$

but finding $\frac{dv}{dx}$ is more involved. It requires application of the chain rule: if $v = v(w)$ where $w = w(x)$, then $\frac{dv}{dx} = \frac{dv}{dw}\frac{dw}{dx}$. We can ignore the constant 1 in $v(x)$, since its derivative is zero and it will thus not affect the final value of $\frac{dv}{dx}$. Thus, let $v(w) = -e^{-\beta w}$ where $w = w(x) = \log(x)$. Then,

$$\frac{dv}{dw} = \beta e^{-\beta w}$$

and

$$\frac{dw}{dx} = \frac{1}{x}.$$

Thus,

$$\begin{aligned}\frac{dv}{dx} &= \frac{dv}{dw}\frac{dw}{dx} \\ &= \frac{1}{x}\left(\beta e^{-\beta w}\right) \\ &= \frac{1}{x}\left(\beta e^{-\beta \log(x)}\right).\end{aligned}$$

Substituting $\frac{dv}{dx}$ into the expression of the product rule and simplifying yields the first derivative of this curve, presented above as Equation 8.

$$
\begin{aligned}
\frac{dy}{dx} &= u\frac{dv}{dx} + v\frac{du}{dx} \\
&= \lambda\left(\frac{1}{x}\left(\beta e^{-\beta\log(x)}\right)\right) + 0\left(1 - e^{-\beta\log(x)}\right) \\
&= \frac{\lambda}{x}\left(\beta e^{-\beta\log(x)}\right)
\end{aligned}
$$

Next, I show how I arrived at Equation 9 by setting the first derivative equal to c, the desired near-zero slope of the entropy curve, and solving for x.

$$
\begin{aligned}
c &= \frac{\lambda}{x}\left(\beta e^{-\beta\log(x)}\right) \\
\frac{c}{\beta e^{-\beta\log(x)}} &= \frac{\lambda}{x} \\
\frac{cx}{\beta e^{-\beta\log(x)}} &= \lambda
\end{aligned}
$$

At this point, taking the logarithm of both sides gives convenient access to the components of the fraction on the left-hand side.

$$
\begin{aligned}
\log\left(\frac{cx}{\beta e^{-\beta\log(x)}}\right) &= \log(\lambda) \\
\log(cx) - \left(\log(\beta) + \log\left(e^{-\beta\log(x)}\right)\right) &= \log(\lambda) \\
\log(cx) - (\log(\beta) - \beta\log(x)) &= \log(\lambda) \\
\log(c) + \log(x) - \log(\beta) + \beta\log(x) &= \log(\lambda)
\end{aligned}
$$

Next, I group all terms containing x and simplify both sides.

$$\log(x) + \log(x^{\beta}) = \log(\lambda) + \log(\beta) - \log(c)$$
$$\log(x^{\beta}x) = \log\left(\frac{\lambda\beta}{c}\right)$$
$$\log(x^{\beta+1}) = \log\left(\frac{\lambda\beta}{c}\right)$$

Finally, exponentiating both sides allows us to isolate x, yielding the equation presented above as Equation 9.

$$x^{\beta+1} = \frac{\lambda\beta}{c}$$
$$x = \left(\frac{\lambda\beta}{c}\right)^{\frac{1}{\beta+1}}$$

For a review of the product rule and chain rule, see Jordan & Smith (2008: 83–87), and for a review of the properties of logarithms, see Jordan & Smith (2008: 33–34).