Constraints on Generalization: Why are Past-Tense Irregularization Errors so Rare?

Niels Taatgen (niels@ai.rug.nl)
Department of Artificial Intelligence, Grote Kruisstraat 2/1
9712 TS Groningen, Netherlands

Mariëtte Dijkstra (m.e.dijkstra@student.rug.nl)
Departments of Humanities Computing and English, Oude Kijk in 't Jatstraat 26
9712 EK Groningen, Netherlands

Abstract

We present an extension of the ACT-R model by Taatgen and Anderson (2002) of learning the English past tense that can take into account the production of irregularization errors. These errors are produced by using examples of irregular verbs as the basis for analogy. The relative rareness of irregularization errors puts constraints on the rule generalization process. The model explains this by the fact that the probability for a particular irregularization is too low to establish a rule.

Introduction

Learning the past tense in English is an example of generalization in language acquisition that has been a central subject in the debate between symbolism and connectionism. The phenomenon can be described easily: the past tense of regular verbs in English can be obtained by adding *–ed* to the stem. Irregular verbs are unsystematic: each word has a unique inflection. When children learn the inflection of the past tense, they go through three stages. During the first stage, when children first start to use the past tense, they inflect irregular verbs correctly or not at all. During the second stage, they start overregularization of irregular verbs. Although they might have used *broke* earlier on, they now also use **breaked*. The second stage is also characterized by an increase in correct inflection of regular verbs. Once in the third stage all verbs are inflected correctly. The third stage is more approximate: even adults occasionally make overregularization errors. The pattern of learning in the past tense is usually referred to as U-shaped learning.

The motivation to study learning the English past tense, or learning inflection in general, is that it is a clear and well documented example of generalization in language. Somehow the generality of a grammar rule is captured by the cognitive system, and in the process some over-application occurs. The challenge of a model of this generalization process is to capture as many of the details of the learning process as possible.

In order to better compare models of irregular inflection, Marcus (2001) has formulated three criteria in addition to the U-shaped performance curve:

1. The model should be able to freely add *–ed* to words of virtually any sound, for example in *Yeltsin outgorbacheved Gorbachev*.
2. The model should also be able learn regular inflection in cases where regular inflection has a low frequency, as in the German plural, where less than 10 percent of inflection is using the regular *–s* suffix.
3. The model should avoid making other errors than overregularization.

Criterion 3 is based on a study by Xu and Pinker (1995) that investigated what other errors children might make, except for overregularization errors. They made the following classification:

- Over-application of irregular forms, such as *bring-*brang* in analogy with *sing-sang*.
- Blends, where a vowel-change is combined with the regular suffix, such as *sing-*sanged*.
- Gross distortions, where the past tense is an almost unrecognizable form of the stem, as in *mail-*membled.*

The conclusion of the study was that these errors are very rare (they found an error rate of 0.0019), and that gross distortions do not occur at all.

The significance of the low rate of errors other than overregularization lies in the fact that it demonstrates that human generalization is quite sophisticated. Apparently only generalizations are attempted that make some sense, and are rooted in experience. As a consequence, we have to rule out mechanisms for generalization that propose a rather liberal mechanism of producing generalizations, and rely on a feedback or a utility mechanism to weed out improper generalizations.

Theories and Models of Learning the Past Tense

The central debate around learning the past tense focuses on whether a single or a dual system of representation is needed to explain the data. The dual representation theory, best explained by Marcus et al. (1992), uses rules and examples to explain the phenomenon. In this explanation, past tenses are memorized as separate cases, until in stage 2 a general rule is learned, the regular rule (add *–ed* to the stem).

From then on the system first attempts to retrieve a past tense from memory, and if this fails uses the regular rule. As more past tenses are properly memorized, the number of overregularization errors decreases.

Single representation theory, first expressed in the Rumelhart and McClelland model of the past tense (1986), states that only a single representation is needed to model the phenomenon: a neural network. Their association network was able to learn inflection of the past tense, while at the same time exhibiting a U-shape in performance on irregular verbs. Their explanation, and that of many subsequent neural networks (e.g., Plunkett & Marchman, 1991), states that initially the neural network is able to accommodate all individual examples, but as the size of the vocabulary increases, the network is forced to generalize, producing decreased performance on irregular verbs. However, this reliance on vocabulary growth and constitution is also the Achilles heel of the model: the input of the model has to be carefully controlled in order to achieve the desired performance. For example, Rumelhart and McClelland increased their vocabulary from ten to 420 words just before the onset of the decrease in performance on irregular verbs. More modern neural network models have more moderate schemes of increasing vocabulary, but almost always with some growth spurt. Another problem is the constitution of the vocabulary: despite the fact that regular verbs far outnumber irregular verbs, the actual use of irregular verbs is around 70% (the so-called token-frequency). If this raw input of 70% irregular verbs is presented to a neural network, it cannot discover the regularity (Marcus's criterion 2 refers to this problem). With respect to errors, neural network models tend to produce many different types: overregularizations, irregularizations, blends, and other unaccountable errors, thereby violating Marcus's criterion 3. A final problem, notably for the more advanced three-layer backpropagation networks, is that they require feedback on their own production, despite the well-known fact that parents do not consistently give feedback on syntactic errors.

The pattern that emerges from the problems that neural network models have is that they are underconstrained with respect to generalization. Their learning characteristics are too much determined by the input, producing a mismatch between learning in networks and actual human learning. In the remainder of the paper, we will present an alternative model using the ACT-R cognitive architecture. This model extends and earlier model by Taatgen and Anderson (2002; Taatgen, 2001) by using a phonological instead of a symbolic representation of the vocabulary. This extension allows the study of errors other than overregularization errors.

ACT-R

The basic theoretical foundation of the ACT-R (Anderson & Lebiere, 1998) architecture is *rational analysis*. According to rational analysis, each component of the cognitive system is optimized with respect to demands from the environment, given its computational limitations. The main components in ACT-R are a declarative (fact) memory and a production (rule) memory. To avoid confusion with grammatical rules, we will refer to rules in production memory with *production rules*. ACT-R is a so-called hybrid architecture, in the sense that it has both symbolic and sub-symbolic aspects. We will introduce these components informally.

Items in declarative memory, called *chunks*, have different levels of *activation* to reflect their use: chunks that have been used recently or chunks that are used very often receive a high activation. This activation decays over time if the chunk is not used. Activation represents the probability (actually, the log odds) that a chunk is needed and the estimates provided for by ACT-R's learning equations represent the probabilities in the environment very well. The level of activation has a number of effects. One effect of activation is that when ACT-R can choose between chunks, it will retrieve the chunk with the highest activation. Activation also affects retrieval time, and whether the chunk can be retrieved at all.

Chunks cannot act by themselves, they need *production rules* for their application. In order to use a chunk, a production rule has to be invoked that retrieves it from declarative memory and does something with it. Since ACT-R is a goal-driven theory, chunks are usually retrieved to achieve some sort of goal. In the context of learning past tense the goal is simple: given the stem of a word, produce the past tense.

The behavior of production rules is also governed by the principle of rational analysis. Each production rule has a real-value quantity associated with its utility. This utility is calculated from estimates of the cost and probability of reaching the goal if that production rule is chosen. The unit of cost in ACT-R is time. ACT-R's learning mechanisms constantly update these estimates based on experience. If multiple production rules are applicable for a certain goal, the production rule is selected with the highest expected outcome.

In both declarative and procedural memory, selections are made on the basis of some evaluation, either activation or utility. This selection process is noisy, so the item with the highest value has the greatest probability of being selected, but other items get opportunities as well. This may produce errors or suboptimal behavior, but also allows the system to explore knowledge and strategies that are still evolving.

In addition to the learning mechanisms that update activation and expected outcome, ACT-R can also learn new chunks and production rules. New chunks are learned automatically: each time a goal is completed it

is added to declarative memory. If an identical chunk is already present in memory, both chunks are merged and their activation values are combined. New production rules are learned on the basis of specializing and merging existing production rules. The circumstance for learning a new production rule is that two rules fire one after another with the first rule requesting a chunk from memory, and the second rule matching that chunk. A new production rule is formed that combines the two into a macro-rule but eliminates the retrieval. The macro-rule is specialized to contain the information that was retrieved. New rules are not allowed to compete with old rules right away: they are initially introduced with a low utility value[1]. This utility value is increased each time the rule recreated, until at some point the rule is chosen, and its utility estimate can be based on real experience with the rule.

Generalization in ACT-R

The production compilation mechanism in ACT-R only produces more specialized versions of existing rules. How can it then achieve generalization? We will look at examples later, but the general idea is as follows. The key to generalization of an example is that it can be accomplished by specializing even more general rules. A very useful strategy that can serve as a basis for this is analogy. Analogy tries to solve a problem by retrieving an example of a similar problem from memory. This example is then mapped onto the present problem, and the resulting pattern is used to solve the current problem. If production compilation is applied to the process of analogy, the retrieval of the example is substituted into the rules themselves, and this produces a specialized rule that is a generalization of the example. In order for this rule to play a role in the system, it has to be recreated a number of times, and this will happen more often if different combinations of problem and example will produce the same generalization. This property will be crucial in explaining the error patterns in the past tense.

A Model of Learning the Past Tense in ACT-R

The ACT-R model is a dual-representation model that uses ACT-R's declarative memory to store examples of past tenses, and that learns a production rule for the regular past tenses. An important assumption of the model is that irregular past tenses have some sort of advantage over regular past tenses, either because regular past tenses are slightly longer, because a phoneme or syllable is added instead of vowel-change, or because regular past tenses are phonetically irregular (Burzio, 2002). Based on that assumption, ACT-R's theory of rational analysis can explain why irregular past tenses make up for the words with the highest frequency: high frequency words are used often, so storing an irregular past tense separately pays off, while low-frequency past tenses are better inflected through a regular rule, where one rule can cover all cases. Because the frequency distribution of past tenses concurs so well with what one would expect on the basis of the ACT-R theory, the model is fairly straightforward.

Representation

In contrast with earlier ACT-R models (Taatgen & Anderson, 2002; Taatgen, 2001) we will use a phonological representation of the verbs. Table 1 illustrates the contrast between the two representations with an example of the past tense of *believe*. The representation is limited to two syllables. In the 478 verb vocabulary we use, there is only one three-syllable word (which is therefore excluded), so this limitation is of no consequence for the results. For the remainder of this paper, we will, for the sake of brevity, pretend that words are all monosyllabic, and have the structure onset-peak-coda-suffix. The model, however, implements two syllables.

Table 1: Example of the old and the new representation of the verb *believe.*

Old representation	New representation
Word322	Word322
isa past-tense	isa past-tense
word believe	onset1 b
past-stem believe	peak1 schwa
past-suffix ed	coda1 blank
	onset2 l
	peak2 ii
	coda2 v
	past-onset1 b
	past-peak1 schwa
	past-coda1 blank
	past-onset2 l
	past-peak2 ii
	past-coda2 v
	past-suffix d

Basic Strategies

The *retrieval strategy* is to search declarative memory for an example that is identical to the current problem. This retrieved example immediately provides the (hopefully correct) answer. The *analogy strategy* also searches declarative memory for an example, but not necessarily identical to the current problem. Pattern matching rules try to find a pattern in the example that can be applied to the current problem. Analogy is not necessarily successful, as it can find examples that cannot be mapped on the current problem, or it can make a mapping that produces a wrong answer. If both

[1] This particular mechanism is not part of the standard ACT-R distribution, see Taatgen (2002) for details.

strategies fail, the model will use the present tense as past tense.

Because the analogy strategy is the key to the explanation of errors, we will examine it in more detail. Analogy in this model is a two-step process:

1. Retrieve an example similar to the current problem
2. Find the pattern in the example and apply it to the current problem

Although only one rule is needed for step 1, a set of rules is necessary for step 2, one for each possible combination of patterns in problem and example. In the model this means we have the following rule for retrieval[2]:

Retrieve-for-Analogy:
IF the goal is to find inflection *inflection* of a word
THEN make a retrieval request for an example of inflection *inflection*

Although this rule retrieves an arbitrary past tense, ACT-R's activation mechanism ensures that it is likely that a high-frequency verb with many phonemes in common with the current word will be found. Also note that this rule is not specific to the past tense: it can be used for any type of inflection. For the rules in step 2 we have included the following rules:

Analogy-1
IF the goal is to find inflection *inflection* of the word *onset1-peak1-coda1*
AND an example has been retrieved of the word *onset2-peak2-coda2* with *inflection onset2-peak2-coda2-suffix*
THEN set the *inflection* to *onset1-peak1-coda1-suffix*

This rule serves as the basis for the regular rule: if it retrieves an example in which a certain suffix is added to the stem, as is the case with all regular past tenses, it will also add this suffix to the current verb.

Analogy-2:
IF the goal is to find inflection *inflection* of the word *onset1-peak1-coda1*
AND an example has been retrieved of the word *onset2-peak1-coda1* with *inflection onset2-peak2-coda1-suffix*
THEN set the *inflection* to *onset1-peak2-coda1-suffix*

This rule produces vowel-changes: if it retrieves an example with the same peak and coda as the current verb, it will apply the same vowel change (assuming there is one) to the current verb. This rule would, for example, given the example *sing-sang*, inflect *bring* to **brang*. Plain and simple: **Analogy-2** applies if the current verb and the example rhyme.

Analogy-3
IF the goal is to find inflection *inflection* of the word *onset1-peak1-coda1*
AND an example has been retrieved of the word *onset1-peak1-coda2* with *inflection onset1-peak2-coda2-suffix*
THEN set the *inflection* to *onset1-peak2-coda1-suffix*

This rule is the reverse of **Analogy-2**: it applies a vowel change whenever the onset and peak of the example and current verb match.

In the present model we didn't introduce any other analogy rules, but one can image that an even larger set of rules exists. Another aspect of the analogy rules is that each of them consists of a combination of copy and substitute operations. As such they themselves are the result of a compilation process of smaller operations, possibly tuned to analogy patterns that occur often in language.

Examples of Production Compilation

As explained before, the production compilation mechanism merges two rules into one while substituting the retrieved chunk from declarative memory. Given the rules above, the **Retrieve-for-Analogy** rule can be combined with either Analogy rule and an example to form a new rule. For example, **Retrieve-for-Analogy**, **Analogy-1** and the example *work-worked* will produce the regular rule:

Learned-rule-1
IF the goal is to find the past tense of the verb *onset1-peak1-coda1*
THEN set the past tense to *onset1-peak1-coda1*-ed

Similarly, **Retrieve-for-Analogy**, **Analogy-2** and the example *sing-sang* (and given that the current verb also ends in *–ing*) will produce the following rule:

Learned-rule-2
IF the goal is to find the past tense of the verb *onset1*-i-ng-blank
THEN set the past tense to *onset1*-a-ng-blank

During the model simulation, the model will also learn rules based on the direct retrieval strategy. These rules are specialized for specific words, for example:

Learned-rule-3
IF the goal is to find the past tense of the verb b-ii-blank
THEN set the past tense to w-o-z-blank

[2] Note that we use an informal notation of production rules. Terms in *italics* indicate variables.

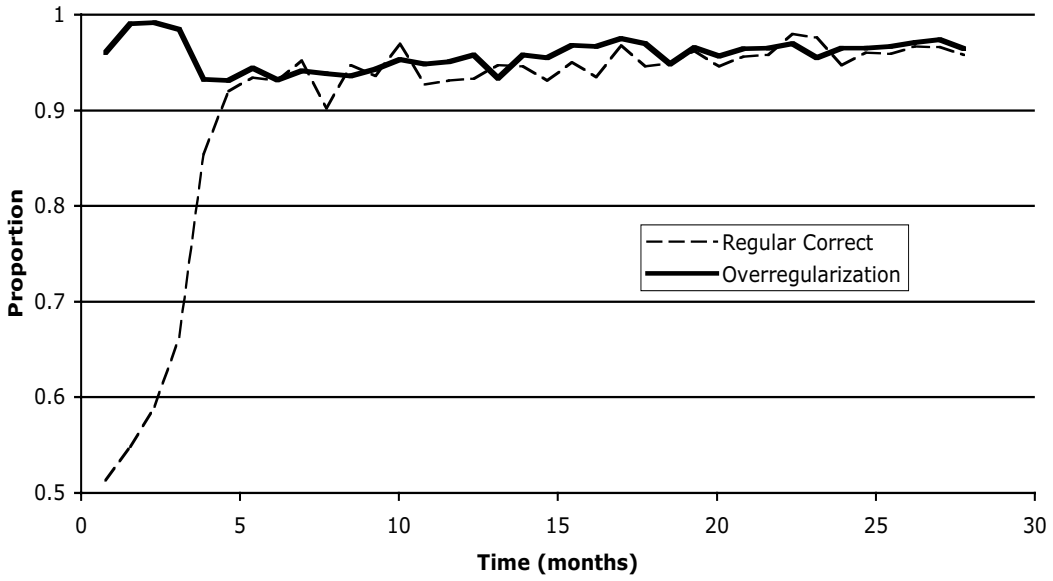

Figure 1. Overregularization and regular correct for the model

Model Simulation

The input for the model consisted of 477 words (478 verbs minus one three-syllable word) from Marcus et al. (1992). Every 2000 simulated seconds two verbs with their past tense were added to declarative memory. This represents the past tenses the child hears in its environment. Also every 2000 simulated seconds one verb was selected for production: the word was presented to the model with the goal to find its past tense. Both the added and presented words were randomly drawn from the vocabulary, but based on the frequency distribution of the words according to Francis and Kucera (1982). The model received no external feedback on its own production, but it did have internal feedback: the amount of time it took to produce a past tense. Reflecting the assumption that irregular past tenses have some advantage, regular inflection received a 400 ms extra cost, and non-inflection received a 600 ms extra cost (the assumption in the case of non-inflection is that past tense must be indicated in another way, for example by adding a word like *yesterday*).

Figure 1 shows the results of 27 simulated months. What is indicated with overregularization is actually the proportion of correct irregular past tenses excluding non-inflection errors. The results clearly show a U-shape in performance, and the onset of overregularization coincides with the a strong improvement in performance on regular verbs. As such, the results are quite comparable to for example Adam (Figure 2, adapted from Marcus et al, 1992). The number of irregularization errors the model makes is so small that it can hardly be plotted: the model made 10 such errors in 35000 inflections, amounting to .029% irregularization errors. The errors the model made varied between runs and very much depended on the combinations of current verb and example. Table 2 shows the errors made in this single run (less than 10 because some errors were duplicated).

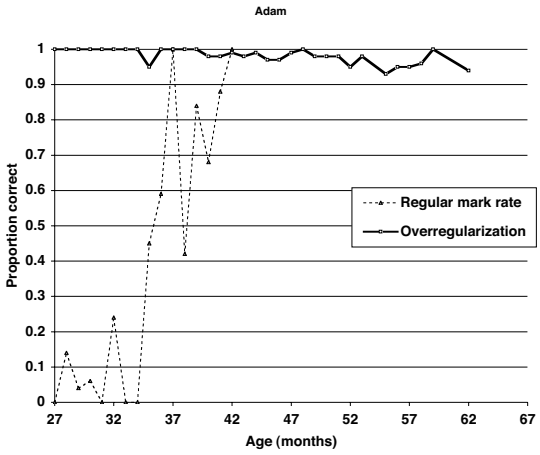

Figure 2. Overregularization and regular correct for Adam

Table 2. Irregularization errors made by the model.

Error	Probably based on
let–*lot	get–got
ride–*rid	hide-hid
bring-*brang	sing-sang
make-*mook	take-took
carve-*ceerve	come-came
feel-*fel	feed-fed

It is clear that the model makes plausible errors, and at least includes the prototypical *bring-*brang*. But why are these errors so much less frequent than over-regularization errors? To understand this, we have to go through the model's functioning in some more detail.

Initially, past tenses are either retrieved, or an attempt at analogy is made. Whenever either retrieval or analogy fails, the other strategy (retrieval or analogy) is tried, and after both have failed the model doesn't inflect at all. Retrieval is only successful when the answer can be found in declarative memory. As the model gets more and more examples from the outside world, it will get more and more successful. Analogy never has problems retrieving examples: its problem is to find a mapping between the retrieved example and the current verb. This will fail most of the time. For example, the example *be-was* is retrieved fairly often, as it is the most frequent verb and therefore has a high activation, but is virtually useless for analogy. However, any retrieved regular verb will serve as a basis for **Analogy-1** and will produce the regular rule. As it takes some time before the regular rule is actually learned (remember it takes several re-creations of the rule), all the overregularizations in the first few months of the simulation are due to the application of the analogy strategy. Only later (around month 4) has the regular rule's utility become high enough to be able to compete with the basic strategies.

The irregularization errors are all produced by **Analogy-2** and **Analogy-3**. In making these errors, rules are learned that correspond to these errors, so in producing *bring-*brang*, the model learned **Learned-rule-2**. However, this rule, and neither any of the similarly learned rules, is ever recreated again. The pattern is just not frequent enough to make a viable rule in the system. So, all irregularization errors are produced by accidental combinations of current verb and example in the analogy strategy, but are never frequent enough to produce a rule that can compete.

Conclusions

An important criterion for a system that generalizes rules out of examples is that it should produce the right rules, and also that it avoids producing the wrong rules. Only a sufficiently constraint generalization system is able to avoid the latter part of the constraint. In this paper we showed that the ACT-R strategy of specializing the Analogy into a generalization of examples is a candidate for such a system. It also concurs with earlier ideas by Anderson (1987), and also has been successfully used to model completely different types of tasks, like Air Traffic Control (Taatgen & Lee, in press).

The symbolic ACT-R model already addressed Marcus's criteria 1 and 2, and with this model we showed the type of explanation needed for criterum 3. An error type that hasn't been addressed by this model though are the so-called blends, like *sing-*sanged.* A possible explanation for this kind of error is that the child mistakenly believes that *sang* is a present tense. The error can then be explained by the application of the regular rule. Modeling these kinds of errors is certainly not impossible, but would require a slightly broader scope of representation of the vocabulary.

References

Anderson, J.R. (1987). Skill acquisition: compilations of weak-method problem solutions. *Psychological Review, 94*, 192-210.

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Erlbaum.

Burzio, L. (2002). Missing players: Phonology and the Past-tense Debate. *Lingua, 112*, 157-199.

Francis, W. N., & Kucera, H. (1982). *Frequency analysis of English usage: lexicon and grammar.* Boston, MA: Houghton Mifflin.

Marcus, G.F. (2001). *The Algebraic Mind.* Cambridge, MA: MIT-Press.

Marcus, G. F., Pinker, S., Ullman, M., Hollander, M., Rosen, T. J., & Xu, F. (1992). Overregularization in language acquisition. *Monographs of the society for research in child development, 57*(4), 1-182.

Plunkett, K., & Marchman, V. (1991). U-shaped learning and frequency effects in a multi-layered perceptron: Implications for child language acquisition. *Cognition, 38*, 43-102.

Rumelhart, D.E., & McClelland, J.L. (1986). On learning the past tense of English verbs. In J.L. McClelland & D.E. Rumelhart (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (pp. 216-271). Cambridge, MA: MIT Press.

Taatgen, N.A. (2001). Extending the past-tense debate: a model of the German plural. In: *Proceedings of the twenty-third annual conference of the cognitive science society* (pp. 1018-1023). Mahwah, NJ: Erlbaum.

Taatgen, N.A. & Anderson, J.R. (2002). Why do children learn to say "Broke"? A model of learning the past tense without feedback. *Cognition, 86*(1), 123-155.

Taatgen, N.A. & Lee, F.J. (in press). Production compilation: a simple mechanism to model complex skill acquisition. *Human Factors*.

Xu, F. & Pinker, S. (1995). Weird past tense forms. *Journal of child language, 22*, 531-556.