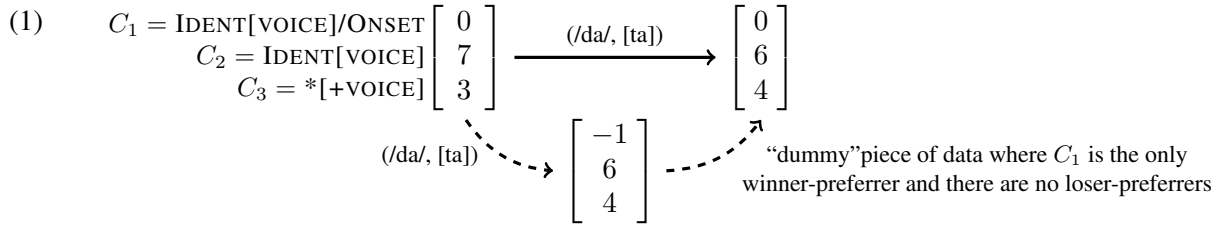
**Error-driven learning in Harmonic Grammar: not impossible, but definitely not optimal**

Giorgio Magri | CNRS and UiL-OTS | magrigrg@gmail.com

■ §**1 -** The HG error-driven learning literature has adopted the *Perceptron* reweighing rule because of its convergence guarantees (e.g., Jesney & Tessier 2011; Coetzee & Pater 2008, 2011; Coetzee & Kawahara 2013; Boersma & Pater 2013). Yet, this rule is not suited to HG, as it fails at ensuring non-negativity of the weights. To solve this impasse, I consider a variant of the Perceptron which truncates any update at zero, thus maintaining the weights non-negative in a principled way. And I show that the convergence guarantees for the original Perceptron extend to its truncated variant (§**2**). Unfortunately, although convergent, the HG error-driven (original and truncated) Perceptron is not efficient, contrary to OT error-driven learners (Tesar & Smolensky 1998; Magri 2012). Indeed, I provide a counterexample with just ten constraints where the HG learner makes over five million errors while the OT learner makes less than 50 errors, and yet the HG and OT typologies coincide! (§**3**). The superiority of OT over HG error-driven learning is shown to extend to the stochastic implementation (§**4**). These results do not contradict the good performance of the Perceptron reported in the Machine Learning (e.g., Mohri *et al.* 2012) and Computational Linguistics (e.g., Collins 2002) literature, as the latter literature has focused on an implementation of the Perceptron which is **not** error-driven (the so called *kernel dual Perceptron*), precisely to cope with the inefficiency of the error-driven Perceptron adopted in the current HG literature.

■ §**2 -** The HG error-driven learner with the original Perceptron reweighing rule decreases the weights of loser-preferring constraints. Hence, weights cannot be guaranteed to be non-negative, as instead required by HG. Even if the weights are initially very large, there is no guarantee that they will not drop below zero, as the number of updates (and thus in particular the number of demotions) crucially depends on the size of the initial weights. To solve this impasse, I consider a variant whereby a weight is decreased *only as long as it does not become negative and it is otherwise not modified*. Is there any way to extend the convergence guarantees of the original Perceptron to this truncated variant? As an example, suppose that the constraint set consists of the three constraints $C_1$, $C_2$, and $C_3$ listed in (1); that the current weights are 0, 7, and 3; that the HG learner is fed the underlying form /da/ together with the corresponding intended winner [ta]; that the only available loser is the faithful [da]. The update performed by the new truncated Perceptron is described by the horizontal arrow in (1): although both $C_1$ and $C_2$ should be demoted because loser-preferring, $C_1$ is not demoted, in order to prevent its weight from turning negative. Crucially, this update can be mimicked with two updates by the original Perceptron: the update triggered by the same piece of data (represented by the left dashed arrow); followed by the update triggered by a "dummy" piece of data whereby $C_1$ is the only winner-preferring constraint and there are no loser-preferring constraints (represented by the right dashed arrow).

(1)




In general, an update according to the truncated Perceptron can always be mimicked through two updates according to the original Perceptron, namely the update triggered by the actual piece of data followed by another update triggered by a *dummy* piece of data whose only purpose is to undo the illicit demotions that yielded negative weights. As these dummy pieces of data only have winner-preferring constraints, they are always consistent with the actual data. Convergence guarantees for the original Perceptron thus extend to the truncated Perceptron, which qualifies as the first principled HG error-driven learner.

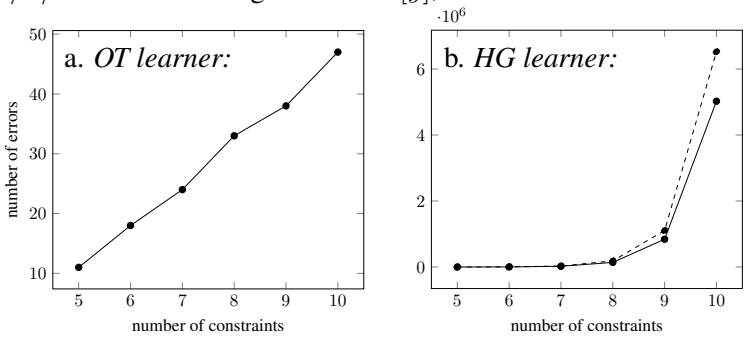
■ §**3 -** Error-driven learning in OT comes with error-bounds that grow slowly (i.e., quadratically) in the number $n$ of constraints for a variety of implementations (Tesar & Smolensky 1998; Magri 2012). I provide a counterexample to show that the number of errors made by HG error-driven learners instead grows very fast (i.e., exponentially) in $n$. The counterexample in the case of $n = 5$ is as follows: there is a unique underlying form $/x/$ with five candidates (called $[y]$ and $[z_1]$, $[z_2]$, $[z_3]$, and $[z_4]$), whose constraint violations are described in (2). The corresponding OT and

(2)

| /x/ | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_5$ |
|---|---|---|---|---|---|
| $[y]$ | | *** | *** | ** | * |
| $[z_1]$ | * | | **** | *** | * |
| $[z_2]$ | | **** | | *** | ** |
| $[z_3]$ | | *** | **** | | ** |
| $[z_4]$ | | *** | *** | *** | |

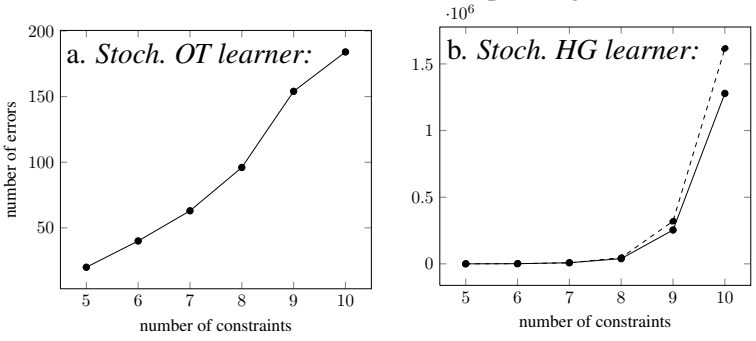
HG typologies (computed with `OT-Help`; Staubs *et al.* 2010) coincide: the mode of constraint interaction has no typological effects in this case. The error-driven learner is trained on the target grammar which maps the underlying form $/x/$ into the winning candidate $[y]$, which therefore beats the four other losing candidates $[z_1]$, $[z_2]$, $[z_3]$, and $[z_4]$. The extension to an arbitrary number $n$ of constraints is straightforward. For each choice of the number $n$ of constrains between 5 and 10, I have run the OT error-driven learner in Magri (2012) ten times on the corresponding counterexample. I plot in (3a) the largest number of errors (vertical axis) performed by the algorithm over the ten runs as a function of the number $n$ of constraints (horizontal axis). The number of errors grows so slowly that only fifty errors are made in the case with $n = 10$ constraints. The solid (dashed) line in (3b) plots the largest number of errors performed by the HG error-driven learner with the original (truncated) Perceptron reweighing rule. The number of errors grows so fast that more than 5 million errors are made already in the case with just $n = 10$ constraints. I will provide a detailed analysis of the bad performance of the HG learner: its number of errors depends on the *margin* of the training data, which I compute analytically for the counterexample considered and show to decrease to zero exponentially fast.

(3)




■ §**4 -** Boersma (1997, 1998) considers a *stochastic* variant of error-driven learning where the current grammar entertained by the learner is stochastically corrupted by additive gaussian noise. Boersma & Pater (2003) consider the HG implementation and show that the convergence guarantees for the Perceptron reweighing rule extend to the stochastic variant. I will show that the same holds for the OT implementation: the convergence guarantees for error-driven ranking algorithms (Tesar & Smolensky 1998, Magri 2012) extend to the stochastic variant. In the worst case, a stochastic learner makes more errors than the corresponding deterministic learner. The crucial question is: how many more additional errors? For the OT implementation, the additional number of errors made by the stochastic variant is proven to grow slowly (i.e., quadratically) in the number $n$ of constraints. The counterexample illustrated above can be sued to show that the situation is different for the HG implementation: the additional number of errors grows very fast (i.e., exponentially) in $n$. For each choice of the number $n$ of constrains between 5 and 10, I have run the deterministic and stochastic OT error-driven learner ten times on the corresponding counterexample described above. I plot in (4a) the difference between the largest number of errors made by the stochastic algorithm over the ten runs minus the number of errors made in that same run by the deterministic algorithm. The plot shows that the additional number of errors due to the stochastic component grows slowly so that less than 200 additional errors are made in the case corresponding to $n = 10$ constraints. The solid (dashed) line in (4b) plots the largest additional number of errors performed by the HG error-driven learner with the original (truncated) Perceptron reweighing rule. The number of additional errors grows so fast that more than 1 million errors are made in the case with just $n = 10$ constraints. I will provide a detailed analysis of the bad performance of the stochastic HG learner: the error bound obtained by Boersma & Pater (2003) for the stochastic learner contains an additional term which depends on the margin and thus grows exponentially fast in the counterexample considered, where the margin of the data decreases exponentially fast.

(4)




■ §**5 -** I conclude that, although error-driven learning is possible in HG (§2), the OT implementation of error-driven learning substantially outperforms the HG implementation from the perspective of worst-case error-bounds, both for the deterministic variant (§3) and the stochastic one (§4).