Generating Contrastive Referring Expressions

Martín Villalba and **Christoph Teichmann** and **Alexander Koller**
Department of Language Science and Technology
Saarland University, Germany
{villalba|cteichmann|koller}@coli.uni-saarland.de

Abstract

The referring expressions (REs) produced by a natural language generation (NLG) system can be misunderstood by the hearer, even when they are semantically correct. In an interactive setting, the NLG system can try to recognize such misunderstandings and correct them. We present an algorithm for generating corrective REs that use contrastive focus (“no, the BLUE button”) to emphasize the information the hearer most likely misunderstood. We show empirically that these contrastive REs are preferred over REs without contrast marking.

1 Introduction

Interactive natural language generation (NLG) systems face the task of detecting when they have been misunderstood, and reacting appropriately to fix the problem. For instance, even when the system generated a semantically correct referring expression (RE), the user may still misunderstand it, i.e. resolve it to a different object from the one the system intended. In an interactive setting, such as a dialogue system or a pedestrian navigation system, the system can try to detect such misunderstandings – e.g. by predicting what the hearer understood from their behavior (Engonopoulos et al., 2013) – and to produce further utterances which resolve the misunderstanding and get the hearer to identify the intended object after all.

When humans correct their own REs, they routinely employ *contrastive focus* (Rooth, 1992; Krifka, 2008) to clarify the relationship to the original RE. Say that we originally described an object b as “the blue button”, but the hearer approaches a button b' which is green, thus providing evidence that they misunderstood the RE to mean b'. In this case, we would like to say “no, the BLUE button”, with the contrastive focus realized by an appropriate pitch accent on “BLUE”. This utterance alerts the hearer to the fact that they misunderstood the original RE; it reiterates the information from the original RE; and it marks the attribute “blue” as a salient difference between b' and the object the original RE was intended to describe.

In this paper, we describe an algorithm for generating REs with contrastive focus. We start from the modeling assumption that misunderstandings arise because the RE r_s the system uttered was corrupted by a noisy channel into an RE r_u which the user “heard” and then resolved correctly; in the example above, we assume the user literally heard “the green button”. We compute this (hypothetical) RE r_u as the RE which refers to b' and has the lowest edit distance from r_s. Based on this, we mark the contrastive words in r_s, i.e. we transform “the blue button” into “the BLUE button”. We evaluate our system empirically on REs from the GIVE Challenge (Koller et al., 2010) and the TUNA Challenge (van der Sluis et al., 2007), and show that the contrastive REs generated by our system are preferred over a number of baselines.

The paper is structured as follows. We first review related work in Section 2 and define the problem of generating contrastive REs in Section 3. Section 4 sketches the general architecture for RE generation on which our system is based. In Section 5, we present the corruption model and show how to use it to reconstruct r_u. Section 6 describes how we use this information to generate contrastive markup in r_s, and in Section 7 we evaluate our approach.

2 Related Work

The notion of focus has been extensively studied in the literature on theoretical semantics and prag-

https://doi.org/10.18653/v1/P17-1063

matics, see e.g. Krifka (2008) and Rooth (1997) for overview papers. Krifka follows Rooth (1992) in taking focus as "indicat(ing) the presence of *alternatives* that are relevant for the interpretation of linguistic expressions"; focus then establishes a contrast between an object and these alternatives. Bornkessel and Schlesewsky (2006) find that corrective focus can even override syntactic requirements, on the basis of "its extraordinarily high communicative saliency". This literature is purely theoretical; we offer an algorithm for automatically generating contrastive focus.

In speech, focus is typically marked through intonation and pitch accents (Levelt, 1993; Pierrehumbert and Hirschberg, 1990; Steube, 2001), while concepts that can be taken for granted are deaccented and/or deleted. Developing systems which realize precise pitch contours for focus in text-to-speech settings is an ongoing research effort. We therefore realize focus in written language in this paper, by capitalizing the focused word. We also experiment with deletion of background words.

There is substantial previous work on interactive systems that detect and respond to misunderstandings. Misu et al. (2014) present an error analysis of an in-car dialogue system which shows that more than half the errors can only be resolved through further clarification dialogues, as opposed to better sensors and/or databases; that is, by improved handling of misunderstandings. Engonopoulos et al. (2013) detect misunderstandings of REs in interactive NLG through the use of a statistical model. Their model also predicts the object to which a misunderstood RE was incorrectly resolved. Moving from misunderstanding detection to error correction, Zarrieß and Schlangen (2016) present an interactive NLG algorithm which is capable of referring in installments, in that it can generate multiple REs that are designed to correct misunderstandings of earlier REs to the same object. The interactive NLG system developed by Akkersdijk et al. (2011) generates both reflective and anticipative feedback based on what a user does and sees. Their error detection and correction strategy distinguishes a fixed set of possible situations where feedback is necessary, and defines custom, hard-coded RE generation sub-strategies for each one. None of these systems generate REs marked for focus.

We are aware of two items of previous work that address the generation of contrastive REs directly. Milosavljevic and Dale (1996) outline strategies for generating clarificatory comparisons in encyclopedic descriptions. Their surface realizer can generate contrastive REs, but the attributes that receive contrastive focus have to be specified by hand. Krahmer and Theune (2002) extend the Incremental Algorithm (Dale and Reiter, 1995) so it can mark attributes as contrastive. This is a fully automatic algorithm for contrastive REs, but it inherits all the limitations of the Incremental Algorithm, such as its reliance on a fixed attribute order. Neither of these two approaches evaluates the quality of the contrastive REs it generates.

Finally, some work has addressed the issue of generating texts that realize the discourse relation *contrast*. For instance, Howcroft et al. (2013) show how to choose contrastive discourse connectives (*but, while, ...*) when generating restaurant descriptions, thus increasing human ratings for naturalness. Unlike their work, the research presented in this paper is not about discourse relations, but about assigning focus in contrastive REs.

3 Interactive NLG

We start by introducing the problem of generating corrective REs in an interactive NLG setting. We use examples from the GIVE Challenge (Koller et al., 2010) throughout the paper; however, the algorithm itself is domain-independent.

GIVE is a shared task in which an NLG system (the instruction giver, IG) must guide a human user (the instruction follower, IF) through a virtual 3D environment. The IF needs to open a safe and steal a trophy by clicking on a number of buttons in the right order without triggering alarms. The job of the NLG system is to generate natural-language instructions which guide the IF to complete this task successfully.

The generation of REs has a central place in the GIVE Challenge because the system frequently needs to identify buttons in the virtual environment to the IF. Figure 1 shows a screenshot of a GIVE game in progress; here b_1 and b_4 are blue buttons, b_2 and b_3 are yellow buttons, and w_1 is a window. If the next button the IF needs to press is b_4 – the *intended object*, o_s – then one good RE for b_4 would be "the blue button below the window", and the system should utter:

(1) Press the blue button below the window.

After uttering this sentence, the system can

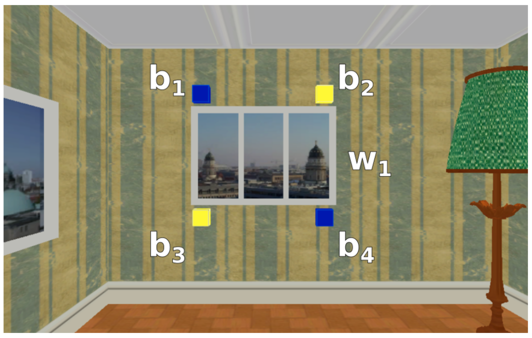


Figure 1: Example scene from the GIVE Challenge.

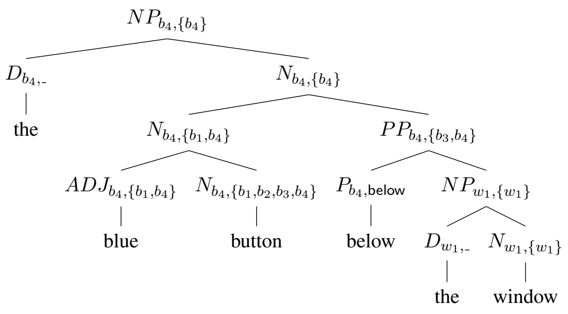


Figure 2: Example syntax tree for an RE for b_4.

track the IF's behavior to see whether the IF has understood the RE correctly. If the wrong button is pressed, or if a model of IF's behavior suggests that they are about to press the wrong button (Engonopoulos et al., 2013), the original RE has been misunderstood. However, the system still gets a second chance, since it can utter a *corrective* RE, with the goal of identifying b_4 to the IF after all. Examples include simply repeating the original RE, or generating a completely new RE from scratch. The system can also explicitly take into account which part of the original RE the IF misunderstood. If it has reason to believe that the IF resolved the RE to b_3, it could say:

(2) No, the BLUE button below the window.

This use of contrastive focus distinguishes the attributes the IF misunderstood (blue) from those that they understood correctly (below the window), and thus makes it easier for the IF to resolve the misunderstanding. In speech, contrastive focus would be realized with a pitch accent; we approximate this accent in written language by capitalizing the focused word. We call an RE that uses contrastive focus to highlight the difference between the misunderstood and the intended object, a *contrastive RE*. The aim of this paper is to present an algorithm for computing contrastive REs.

4 Generating Referring Expressions

While we make no assumptions on how the original RE r_s was generated, our algorithm for reconstructing the corrupted RE r_u requires an RE generation algorithm that can represent all semantically correct REs for a given object compactly in a chart. Here we sketch the RE generation of Engonopoulos and Koller (2014), which satisfies this requirement.

This algorithm assumes a synchronous grammar which relates strings with the sets of objects they refer to. Strings and their referent sets are constructed in parallel from lexicon entries and grammar rules; each grammar rule specifies how the referent set of the parent is determined from those of the children. For the scene in Figure 1, we assume lexicon entries which express, among other things, that the word "blue" denotes the set $\{b_1, b_4\}$ and the word "below" denotes the relation $\{(w_1, b_1), (w_1, b_2), (b_3, w_1), (b_4, w_1)\}$. We combine these lexicons entries using rules such as

"N $\rightarrow$ button() |button |$\{b_1, b_2, b_3, b_4\}$"

which generates the string "button" and associates it with the set of all buttons or

"N $\rightarrow$ N1(N,PP) |$w_1 \bullet w_2$ |$R_1 \cap R_2$"

which states that a phrase of type noun can be combined with a prepositional phrase and their denotations will be intersected. Using these rules we can determine that "the window" denotes $\{w_1\}$, that "below the window" can refer to $\{b_3, b_4\}$ and that "blue button below the window" uniquely refers to $\{b_4\}$. The syntax tree in Fig. 2 represents a complete derivation of an RE for $\{b_4\}$.

The algorithm of Engonopoulos and Koller computes a chart which represents the set of all possible REs for a given set of input objects, such as $\{b_4\}$, according to the grammar. This is done by building a chart containing all derivations of the grammar which correspond to the desired set. They represent this chart as a finite tree automaton (Comon et al., 2007). Here we simply write the chart as a Context-Free Grammar. The strings produced by this Context-Free Grammar are then exactly the REs for the intended object. For example, the syntax tree in Fig. 2 is generated by the parse chart for the set $\{b_4\}$. Its nonterminal symbols consist of three parts: a syntactic category

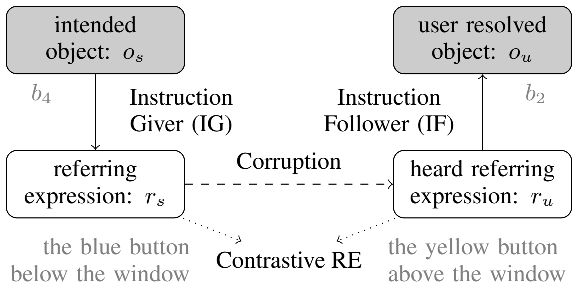

intended object: o_s
user resolved object: o_u
b_4
Instruction Giver (IG)
Instruction Follower (IF)
b_2
referring expression: r_s
Corruption
heard referring expression: r_u
the blue button below the window
Contrastive RE
the yellow button above the window

Figure 3: The corruption model.

(given by the synchronous grammar), the referent for which an RE is currently being constructed, and the set of objects to which the entire subtree refers. The grammar may include recursion and therefore allow for an infinite set of possible REs. If it is weighted, one can use the Viterbi algorithm to compute the best RE from the chart.

5 Listener Hypotheses and Edit Distance

5.1 Corruption model

Now let us say that the system has generated and uttered an RE r_s with the intention of referring to the object o_s, but it has then found that the IF has misunderstood the RE and resolved it to another object, o_u (see Fig. 3). We assume for the purposes of this paper that such a misunderstanding arises because r_s was corrupted by a noisy channel when it was transmitted to the IF, and the IF "heard" a different RE, r_u. We further assume that the IF then resolved r_u correctly, i.e. the corruption in the transmission is the only source of misunderstandings.

In reality, there are of course many other reasons why the IF might misunderstand r_s, such as lack of attention, discrepancies in the lexicon or the world model of the IG and IF, and so on. We make a simplifying assumption in order to make the misunderstanding explicit at the level of the RE strings, while still permitting meaningful corrections for a large class of misunderstandings.

An NLG system that builds upon this idea in order to generate a corrective RE has access to the values of o_s, r_s and o_u; but it needs to infer the most likely corrupted RE r_u. To do this, we model the corruption using the edit operations used for the familiar *Levenshtein edit distance* (Mohri, 2003) over the alphabet Σ: S_a, substitution of a word with a symbol $a \in \Sigma$; D, deletion of a word; I_a, insertion of the symbol $a \in \Sigma$; or K, keeping the word. The noisy channel passes over each word in r_s and applies either D, K or one of the S operations to it. It may also apply I operations before or after a word. We call any sequence s of edit operations that could apply to r_s an *edit sequence for* r_s.

An example for an edit sequence which corrupts r_s = "the blue button below the window" into r_u = "the yellow button above the window" is shown in Figure 4. The same r_u could also have been generated by the edit operation sequence $K\ S_{\text{yellow}}\ K\ S_{\text{above}}\ K\ K$, and there is generally a large number of edit sequences that could transform between any two REs. If an edit sequence s maps x to y, we write apply$(s, x) = y$.

We can now define a probability distribution $P(s \mid r_s)$ over edit sequences s that the noisy channel might apply to the string r_s, as follows:

$$P(s \mid r_s) = \frac{1}{Z} \prod_{s_i \in s} \exp(-c(s_i)),$$

where $c(s_i)$ is a cost for using the edit operation s_i. We set $c(K) = 0$, and for any a in our alphabet we set $c(S_a) = c(I_a) = c(D) = \mathcal{C}$, for some fixed $\mathcal{C} > 0$. Z is a normalizing constant which is independent of s and ensures that the probabilities sum to 1. It is finite for sufficiently high values of $\mathcal{C}$, because no sequence for r_s can ever contain more K, S and D operations than there are words in r_s, and the total weight of sequences generated by adding more and more I operations will converge.

Finally, let L be the set of referring expressions that the IF would resolve to o_u, i.e. the set of candidates for r_u. Then the most probable edit sequence for r_s which generates an $r_u \in L$ is given by

$$\begin{aligned} s^* &= \operatorname*{arg\,max}_{s\,:\,\text{apply}(s,r_s)\in L} P(s \mid r_s) \\ &= \arg\min_s \textstyle\sum_{s_i \in s} c(s_i), \end{aligned}$$

i.e. s^* is the edit sequence that maps r_s to an RE in L with minimal cost. We will assume that s^* is the edit sequence that corrupted r_s, i.e. that r_u = apply(s^*, r_s).

5.2 Finding the most likely corruption

It remains to compute s^*; we will then show in Section 6 how it can be used to generate a corrective RE. Attempting to find s^* by enumeration is impractical, as the set of edit sequences for a given r_s and r_u may be large and the set of possible r_u for a given o_u may be infinite. Instead

r_s	the	blue		button	below	the	window
edit operation sequence	K	D	I_{yellow}	K	S_{above}	K	K
r_u	the		yellow	button	above	the	window

Figure 4: Example edit sequence for a given corruption.

we will use the algorithm from Section 4 to compute a chart for all the possible REs for o_u, represented as a context-free grammar G whose language $L = L(G)$ consists of these REs. We will then intersect it with a finite-state automaton which keeps track of the edit costs, obtaining a second context-free grammar G'. These operations can be performed efficiently, and s^* can be read off of the minimum-cost syntax tree of G'.

Edit automaton. The possible edit sequences for a given r_s can be represented compactly in the form of a weighted finite-state automaton $F(r_s)$ (Mohri, 2003). Each run of the automaton on a string w corresponds to a specific edit sequence that transforms r_s into w, and the sum of transition weights of the run is the cost of that edit sequence. We call $F(r_s)$ the *edit automaton*. It has a state q_i for every position i in r_s; the start state is q_0 and the final state is $q_{|r_s|}$. For each i, it has a "keep" transition from q_i to q_{i+1} that reads the word at position i with cost 0. In addition, there are transitions from q_i to q_{i+1} with cost $\mathcal{C}$ that read any symbol in Σ (for substitution) and ones that read the empty string ϵ (for deletion). Finally, there is a loop with cost $\mathcal{C}$ from each q_i to itself and for any symbol in Σ, implementing insertion.

An example automaton for r_s = "the blue button below the window" is shown in Figure 5. The transitions are written in the form $\langle$word in w : associated cost$\rangle$. Note that every path through the edit transducer corresponds to a specific edit sequence s, and the sum of the costs along the path corresponds to $-\log P(s \mid r_s) - \log Z$.

Combining G and $F(r_s)$. Now we can combine G with $F(r_s)$ to obtain G', by intersecting them using the Bar-Hillel construction (Bar-Hillel et al., 1961; Hopcroft and Ullman, 1979). For the purposes of our presentation we assume that G is in Chomsky Normal Form, i.e. all rules have the form $A \to a$, where a is a word, or $A \to B\ C$, where both symbols on the right hand side are nonterminals. The resulting grammar G' uses nonterminal symbols of the form $N_{b,A,\langle q_i,q_k\rangle}$, where b, A are as in Section 4, and q_i, q_k indicate that the string derived by this nonterminal was generated by editing the substring of r_s from position i to k.

Let $N_{b,A} \to a$ be a production rule of G with a word a on the right-hand side; as explained above, b is the object to which the subtree should refer, and A is the set of objects to which the subtree actually might refer. Let $t = q_i \to \langle a{:}c\rangle q_k$ be a transition in $F(r_s)$, where q, q' are states of $F(r_s)$ and c is the edit cost. From these two, we create a context-free rule $N_{b,A,\langle q_i,q_k\rangle} \to a$ with weight c and add it to G'. If $k = i + 1$, these rules represent K and S operations; if $k = i$, they represent insertions.

Now let $N_{b,A} \to X_{b_1,A_1}\, Y_{b_2,A_2}$ be a binary rule in G, and let q_i, q_j, q_k be states of $F(r_s)$ with $i \le j \le k$. We then add a rule $N_{b,A,\langle q_i,q_k\rangle} \to X_{b_1,A_1,\langle q_i,q_j\rangle}\, Y_{b_2,A_2,\langle q_j,q_k\rangle}$ to G'. These rules are assigned weight 0, as they only combine words according to the grammar structure of G and do not encode any edit operations.

Finally, we deal with deletion. Let $N_{b,A}$ be a nonterminal symbol in G and let q_h, q_i, q_j, q_k be states of $F(r_s)$ with $h \le i \le j \le k$. We then add a rule $N_{b,A,\langle q_h,q_k\rangle} \to N_{b,A,\langle q_i,q_j\rangle}$ to G'. This rule deletes the substrings from positions h to i and j to k from r_s; thus we assign it the cost $((i - h) + (k - j))\mathcal{C}$, i.e. the cost of the corresponding ϵ transitions.

If the start symbol of G is $S_{b,A}$, then the start symbol of G' is $S_{b,A,\langle q_0,q_{|r_s|}\rangle}$. This construction intersects the languages of G and $F(r_s)$, but because $F(r_s)$ accepts all strings over the alphabet, the languages of G' and G will be the same (namely, all REs for o_u). However, the weights in G' are inherited from $F(r_s)$; thus the weight of each RE in $L(G')$ is the edit cost from r_s.

Example. Fig. 6 shows an example tree for the G' we obtain from the automaton in Fig. 5. We can read the string w = "the yellow button above the window" off of the leaves; by construction, this is an RE for o_u. Furthermore, we can reconstruct the edit sequence that maps from r_s to w from the rules of G' that

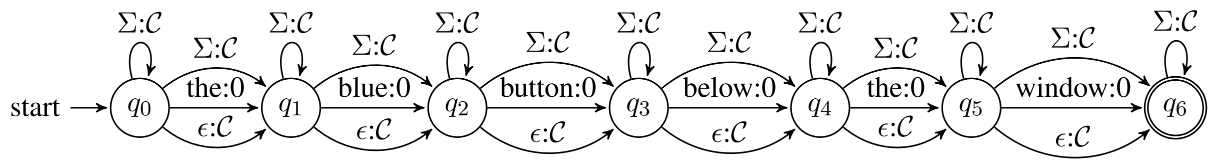


Figure 5: Edit automaton $F(r_s)$ for $r_s =$ "the blue button below the window".

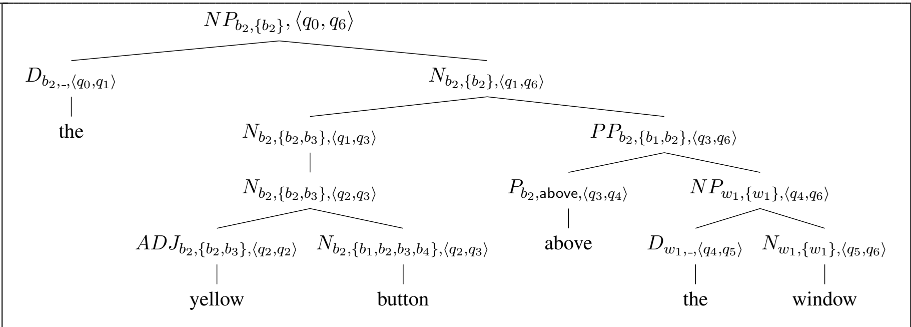
Tree	
s	$K\ D\ I_{\text{yellow}}\ K\ S_{\text{above}}\ K\ K$
Emphasis	No, press the BLUE button BELOW the window

Figure 6: A syntax tree described by G', together with its associated edit sequence and contrastive RE.

were used to derive w. We can see that "yellow" was created by an insertion because the two states of $F(r_s)$ in the preterminal symbol just above it are the same. If the two states are different, then the word was either substituted ("above", if the rule had weight $\mathcal{C}$) or kept ("the", if the rule had weight 0). By contrast, unary rules indicate deletions, in that they make "progress" in r_s without adding new words to w.

We can compute the minimal-cost tree of G' using the Viterbi algorithm. Thus, to summarize, we can calculate s^* from the intersection of a context-free grammar G representing the REs to o_u with the automaton $F(r_s)$ representing the edit distance to r_s. From this, we obtain $r_u = \text{apply}(s^*, r_s)$. This is efficient in practice.

6 Generating Contrastive REs

6.1 Contrastive focus

We are now ready to generate a contrastive RE from r_s and s^*. We assign focus to the words in r_s which were changed by the corruption – that is, the ones to which s^* applied Substitute or Delete operations. For instance, the edit sequence in Fig. 6 deleted "blue" and substituted "below" with "above". Thus, we mark these words with focus, and obtain the contrastive RE "the BLUE button BELOW the window". We call this strategy *Emphasis*, and write ${r_s}^E$ for the RE obtained by applying the Emphasis strategy to the RE r_s.

6.2 Shortening

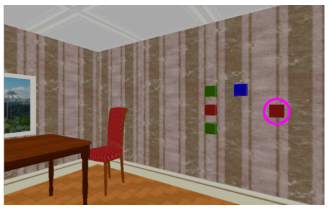
We also investigate a second strategy, which generates more succinct contrastive REs than the *Emphasis* strategy. Most research on RE generation (e.g. Dale and Reiter (1995)) has assumed that hearers should prefer succinct REs, which in particular do not violate the Maxim of Quantity (Grice, 1975). When we utter a contrastive RE, the user has previously heard the RE r_s, so some of the information in ${r_s}^E$ is redundant. Thus we might obtain a more succinct, and possibly better, RE by dropping such redundant information from the RE.

For the grammars we consider here, ${r_s}^E$ often combines an NP and a PP, e.g. "[blue button]$_{NP}$ [below the window]$_{PP}$". If errors occur only in one of these constituents, then it might be sufficient to generate a contrastive RE using only that constituent. We call this strategy *Shortening* and define it as follows.

If all the words that are emphasized in ${r_s}^E$ are in the NP, the Shortening RE is "the" plus the NP, with emphasis as in ${r_s}^E$. So if r_s is "the [blue button] [above the window]" and $s^* = K\ S_{yellow}\ K\ K\ K\ K$, corresponding to a ${r_s}^E$ of "the [BLUE button] [above the window]", then the RE would be "the [BLUE button]".

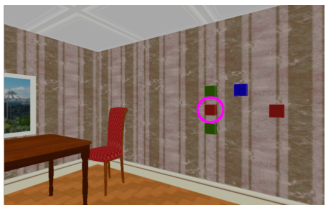
If all the emphasis in ${r_s}^E$ is in the PP, we use

We wanted our player to select this button:



So we told them: *press the red button to the right of the blue button.*

But they selected this button instead:



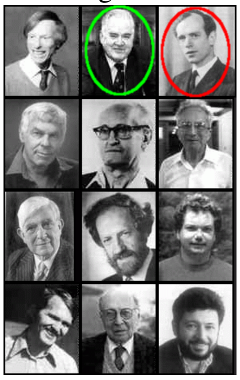
Which correction is better for this scene?

○ No, press the red **BUTTON** to the right of the **BLUE BUTTON**

○ No, press the red button to the **RIGHT** of the blue button

Figure 7: A sample scene from Experiment 1.

We wanted our player to select the person circled in green:



So we told them: *the light haired old man in a suit looking straight.*

But they selected the person circled in red instead. Which correction is better for this scene?

○ No, the light haired old man **IN A SUIT LOOKING STRAIGHT**

○ No, the **LIGHT HAIRED OLD** man in a suit looking straight

Figure 8: A sample scene from Experiment 2.

"the one" plus the PP and again capitalize as in ${r_s}^E$. So if we have $s^* = K\,K\,K\,S_{below}\,K\,K$, where ${r_s}^E$ is "the [blue button] [ABOVE the window]", we obtain "the one [ABOVE the window]." If there is no PP or if ${r_s}^E$ emphasizes words in both the NP and the PP, then we just use ${r_s}^E$.

7 Evaluation

To test whether our algorithm for contrastive REs assigns contrastive focus correctly, we evaluated it against several baselines in crowdsourced pairwise comparison overhearer experiments. Like Buß et al. (2010), we opted for an overhearer experiment to focus our evaluation on the effects of contrastive feedback, as opposed to the challenges presented by the navigational and timing aspects of a fully interactive system.

7.1 Domains and stimuli

We created the stimuli for our experiments from two different domains. We performed a first experiment with scenes from the GIVE Challenge, while a second experiment replaced these scenes with stimuli from the "People" domain of the TUNA Reference Corpus (van der Sluis et al., 2007). This corpus consists of photographs of men annotated with nine attributes, such as whether the person has a beard, a tie, or is looking straight. Six of these attributes were included in the corpus to better reflect human RE generation strategies. Many human-generated REs in the corpus are overspecific, in that they contain attributes that are not necessary to make the RE semantically unique.

We chose the GIVE environment in order to test REs referring both to attributes of an object, i.e. color, and to its spatial relation to other visible objects in the scene. The TUNA Corpus was chosen as a more challenging domain, due to the greater number of available properties for each object on a scene.

Each experimental subject was presented with screenshots containing a marked object and an RE. Subjects were told that we had previously referred to the marked object with the given RE, but an (imaginary) player misunderstood this RE and selected a different object, shown in a second screenshot. They were then asked to select which one of two corrections they considered better, where "better" was intentionally left unspecific. Figs. 7 and 8 show examples for each domain. The full set of stimuli is available as supplementary material.

To maintain annotation quality in our crowdsourcing setting, we designed test items with a clearly incorrect answer, such as REs referring to the wrong target or a nonexistent one. These test items were randomly interspersed with the real stimuli, and only subjects with a perfect score on the test items were taken into account. Experimental subjects were asked to rate up to 12 comparisons, shown in groups of 3 scenes at a time, and were automatically disqualified if they evaluated any individual scene in less than 10 seconds. The order in which the pairs of strategies were shown was randomized, to avoid effects related to the order in which they were presented on screen.

7.2 Experiment 1

Our first experiment tested four strategies against each other. Each experimental subject was presented with two screenshots of 3D scenes with a marked object and an RE (see Fig. 7 for an example). Each subject was shown a total of 12 scenes, selected at random from 16 test scenes. We collected 10 judgments for each possible combination of GIVE scene and pair of strategies, yielding a total of 943 judgements from 142 subjects after removing fake answers.

We compared the *Emphasis* and *Shortening* strategies from Section 6 against two baselines. The *Repeat* strategy simply presented r_s as a "contrastive" RE, without any capitalization. Comparisons to *Repeat* test the hypothesis that subjects prefer explicit contrastive focus. The *Random* strategy randomly capitalized adjectives, adverbs, and/or prepositions that were not capitalized by the *Emphasis* strategy. Comparisons to *Random* verify that any preference for *Emphasis* is not only due to the presence of contrastive focus, but also because our method identifies precisely where that focus should be.

Table 1a shows the results of all pairwise comparisons. For each row strategy $Strat_R$ and each column strategy $Strat_C$, the table value corresponds to

$$\frac{(\#Strat_R \text{ pref. over } Strat_C) - (\#Strat_C \text{ pref. over } Strat_R)}{(\# \text{ tests between } Strat_R \text{ and } Strat_C)}$$

Significance levels are taken from a two-tailed binomial test over the counts of preferences for each strategy. We find a significant preference for the *Emphasis* strategy over all others, providing evidence that our algorithm assigns contrastive focus to the right words in the corrective RE.

While the *Shortening* strategy is numerically preferred over both baselines, the difference is not significant, and it is significantly worse than the *Emphasis* strategy. This is surprising, given our initial assumption that listeners prefer succinct REs. It is possible that a different strategy for shortening contrastive REs would work better; this bears further study.

7.3 Experiment 2

In our second experiment, we paired the *Emphasis*, *Repeat*, and *Random* strategies against each other, this time evaluating each strategy in the TUNA people domain. Due to its poor performance in Experiment 1, which was confirmed in pilot experiments for Experiment 2, the *Shortening* strategy was not included.

The experimental setup for the TUNA domain used 3x4 grids of pictures of people chosen at random from the TUNA Challenge, as shown in Fig. 8. We generated 8 such grids, along with REs ranging from two to five attributes and requiring one or two attributes to establish the correct contrast. The larger visual size of objects in the the TUNA scenes allowed us to mark both o_s and o_u in a single picture without excessive clutter.

The REs for Experiment 2 were designed to only include attributes from the referred objects, but no information about its position in relation to other objects. The benefit is twofold: we avoid taxing our subjects' memory with extremely long REs, and we ensure that the overall length of the second set of REs is comparable to those in the previous experiment.

We obtained 240 judgements from 65 subjects (after removing fake answers). Table 1b shows the results of all pairwise comparisons. We find that even in the presence of a larger number of attributes, our algorithm assigns contrastive focus to the correct words of the RE.

7.4 Discussion

Our experiments confirm that the strategy for computing contrastive REs presented in this paper works in practice. This validates the corruption model, which approximates semantic mismatches between what the speaker said and what the listener understood as differences at the level of words in strings. Obviously, this model is still an approximation, and we will test its limits in future work.

We find that users generally prefer REs with an emphasis over simple repetitions. In the more challenging scenes of the TUNA corpus, users even have a significant preference of Random over

	Repeat	Random	Emphasis	Shortening
Repeat	–	0.041	-0.570***	-0.141
Random	-0.041	–	-0.600***	-0.109
Emphasis	0.570***	0.600***	–	0.376***
Shortening	0.141	0.109	-0.376***	–

(a) Results for Experiment 1

	Repeat	Random	Emphasis
Repeat	–	-0.425***	-0.575***
Random	0.425***	–	-0.425***
Emphasis	0.575***	0.425***	–

(b) Results for Experiment 2

Table 1: Pairwise comparisons between feedback strategies for experiments 1 and 2. A positive value shows preference for the row strategy, significant at *** $p < 0.001$.

Repeat, although this makes no semantic sense. This preference may be due to the fact that emphasizing *anything* at least publically acknowledges the presence of a misunderstanding that requires correction. It will be interesting to explore whether this preference holds up in an interactive setting, rather than an overhearer experiment, where listeners will have to act upon the corrective REs.

The poor performance of the Shortening strategy is a surprising negative result. We would expect a shorter RE to always be preferred, following the Gricean Maxim of Quantity (Grice, 1975). This may because our particular Shortening strategy can be improved, or it may be because listeners interpret the shortened REs not with respect to the original instructions, but rather with respect to a "refreshed" context (as observed, for instance, in Gotzner et al. (2016)). In this case the shortened REs would not be unique with respect to the refreshed, wider context.

8 Conclusion

In this paper, we have presented an algorithm for generating contrastive feedback for a hearer who has misunderstood a referring expression. Our technique is based on modeling likely user misunderstandings and then attempting to give feedback that contrasts with the most probable incorrect understanding. Our experiments show that this technique accurately predicts which words to mark as focused in a contrastive RE.

In future work, we will complement the overhearer experiment presented here with an end-to-end evaluation in an interactive NLG setting. This will allow us to further investigate the quality of the correction strategies and refine the Shortening strategy. It will also give us the opportunity to investigate empirically the limits of the corruption model. Furthermore, we could use this data to refine the costs $c(D), c(I_a)$ etc. for the edit operations, possibly assigning different costs to different edit operations.

Finally, it would be interesting to combine our algorithm with a speech synthesis system. In this way, we will be able to express focus with actual pitch accents, in contrast to the typographic approximation we made here.

References

Saskia Akkersdijk, Marin Langenbach, Frieder Loch, and Mariët Theune. 2011. The thumbs up! twente system for give 2.5. In *The 13th European Workshop on Natural Language Generation (ENLG 2011)*.

Yehoshua Bar-Hillel, Micha Perles, and Eli Shamir. 1961. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung* 14:143–172.

Ina Bornkessel and Matthias Schlesewsky. 2006. The role of contrast in the local licensing of scrambling in german: Evidence from online comprehension. *Journal of Germanic Linguistics* 18(01):1–43.

Okko Buß, Timo Baumann, and David Schlangen. 2010. Collaborating on utterances with a spoken dialogue system using an isu–based approach to incremental dialogue management. In *Proceedings of the Special Interests Group on Discourse and Dialogue Conference (SIGdial 2010)*.

Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Sophie Tison, Marc Tommasi, and Christof Löding. 2007. *Tree Automata techniques and applications*. published online - `http://tata.gforge.inria.fr/`. http://tata.gforge.inria.fr/.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science* 19(2):233–263.

Nikos Engonopoulos and Alexander Koller. 2014. Generating effective referring expressions using charts. In *Proceedings of the INLG and SIGdial 2014 Joint Session*.

Nikos Engonopoulos, Martín Villalba, Ivan Titov, and Alexander Koller. 2013. Predicting the resolution of referring expressions from user behavior. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*.

Nicole Gotzner, Isabell Wartenburger, and Katharina Spalek. 2016. The impact of focus particles on the recognition and rejection of contrastive alternatives. *Language and Cognition* 8(1):59–95.

H. Paul Grice. 1975. Logic and conversation. In P. Cole and J. L. Morgan, editors, *Syntax and Semantics: Vol. 3: Speech Acts*, Academic Press, pages 41–58.

John Edward Hopcroft and Jeffrey Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley.

David Howcroft, Crystal Nakatsu, and Michael White. 2013. Enhancing the expression of contrast in the SPaRKy restaurant corpus. In *Proceedings of the 14th European Workshop on Natural Language Generation (ENLG 2013)*.

Alexander Koller, Kristina Striegnitz, Andrew Gargett, Donna Byron, Justine Cassell, Robert Dale, Johanna Moore, and Jon Oberlander. 2010. Report on the Second NLG Challenge on Generating Instructions in Virtual Environments (GIVE-2). In *Proceedings of the Sixth International Natural Language Generation Conference (Special session on Generation Challenges)*.

E. Krahmer and M. Theune. 2002. Efficient context-sensitive generation of referring expressions. In K. van Deemter and R. Kibble, editors, *Information Sharing: Reference and Presupposition in Language Generation and Interpretation, Center for the Study of Language and Information-Lecture Notes*, CSLI Publications, volume 143, pages 223–263.

Manfred Krifka. 2008. Basic notions of information structure. *Acta Linguistica Hungarica* 55:243–276.

Willem J.M. Levelt. 1993. *Speaking: From Intention to Articulation*. MIT University Press Group.

Maria Milosavljevic and Robert Dale. 1996. Strategies for comparison in encyclopædia descriptions. In *Proceedings of the 8th International Natural Language Generation Workshop (INLG 1996)*.

Teruhisa Misu, Antoine Raux, Rakesh Gupta, and Ian Lane. 2014. Situated language understanding at 25 miles per hour. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGdial 2014)*.

Mehryar Mohri. 2003. Edit-distance of weighted automata: General definitions and algorithms. *International Journal of Foundations of Computer Science* 14(6):957–982.

Janet B. Pierrehumbert and Julia Hirschberg. 1990. The meaning of intonational contours in the interpretation of discourse. In Philip R. Cohen, Jerry Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, MIT University Press Group, chapter 14.

Mats Rooth. 1992. A theory of focus interpretation. *Natural Language Semantics* 1:75–116.

Mats Rooth. 1997. Focus. In Shalom Lappin, editor, *The Handbook of Contemporary Semantic Theory*, Blackwell Publishing, chapter 10, pages 271–298.

Anita Steube. 2001. Correction by contrastive focus. *Theoretical Linguistics* 27(2-3):215–250.

Ielka van der Sluis, Albert Gatt, and Kees van Deemter. 2007. Evaluating algorithms for the generation of referring expressions: Going beyond toy domains. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2007)*.

Sina Zarrieß and David Schlangen. 2016. Easy Things First: Installments Improve Referring Expression Generation for Objects in Photographs. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*.