Formal vs Informal Language

by mav, 2018

Formalists assert that an ideal language should be entirely formal in that the language is strictly explicit and adheres to sentential logic. Informalists believe language is useful for more than just conveying information and that meaning is not dependent on the literal definition or phrasing of words. Traditional formalists and informalists share the belief that formal language fails to convey what British philosopher Paul Grice refers to as "implicatures". Grice disputes neither party individually, opting instead to explain his stance on implicatures regardless of the formality or informality of language.

I support Grice's stance with two arguments of my own: (A) In situations where an absolute formal language exists, it can be "misused" for the sake of conveying the implicit. Implicatures arise naturally in human communication, even when utilizing formal languages. (B) Additionally, I would argue that an informal language, which linguistically allows for implicit statements, is potentially equivalent to a formal language that is lacking the necessary vocabulary for formal versions of the same statements. I will begin by briefly explaining what is meant by the implicature maxims so as to differentiate Grice's ideas from my own. I will explain and concretely support claim (A), but support my claim (B) with somewhat more abstract reasoning.

Grice's implicature elaborates on the implied messages behind what is said, both in proper or conversational communication. He explains his maxims which establish whether or not a given implicature is valid and gives scenarios in which maxims are violated. Where the maxims are adhered to, Grice's illustrations make it clear that it is perfectly reasonable to make relevant and valid conversational implicatures using separate and formally unassociated statements.

For example:

A: *"I am out of petrol"*

B: *"There is a garage around the corner"*

(Grice, *Studies in the Way of Words*, 32)

Though response B seems formally irrelevant to A and does not explicitly satisfy the maxims, it succeeds when we account for the implicit information as a part of the statement (that the garage is open and has petrol for sale).

Conversely, it is possible to violate the maxims while being formally sound:

A: *"How much money did you steal?"*

B: *"I stole 100 dollars"*

In the situation above, person B may have indeed stolen $100, but they also may have stolen an additional $100 that they, for whatever reason, consider to be a distinct group of dollars and have opted to not discuss. This violates the cooperative principle as person B is speaking with the intent to be misunderstood.

There are many languages in existence that are formal by necessity (mathematics, music, etc.). I will be using computer programming languages as my key example for explaining claim (A) as they are formal and culturally relevant. A computer programming language is used by humans as instructive communication with machines to perform any tasks that are reasonably computable. Such languages are entirely and strictly formal; any sort of syntax aberration will result in not just the statement being invalid, but the entire program likely failing its intended task. Despite this, it is still possible to include implicit information within statements that only certain parties are privy to. I am essentially claiming that implicatures, as defined by Grice, are always possible, regardless of the language's extent of formality. A good example is what is referred to as "programmer humor," a niche subgenre of comedy that relies on the implicit understanding of a formal language in order to recognize humorous misuse and inefficiencies or to sympathize with relatable shortcomings.

The following block of code is formatted with generalized and stereotypical programming syntax rather than a specific language and should be clearly legible to users of any modern language (thanks to Reddit user vivlam for the joke idea):

```
1.    function chickenOrEgg()
2.    {
3.          return chicken();
4.    }
5.    function chicken()
6.    {
7.          return egg();
8.    }
9.    function egg()
10.   {
11.         return chicken();
12.   }
13.   print(“the “ + chickenOrEgg() + “ came first”);
```

The 1st, 5th, and 9th lines are all function declaration lines; these lines create a new function and grant it an identity. Everything between the open and close braces { } that follow a function declaration becomes the contents of the function, thus lines 1-12 tell the computer to create three functions and populate them with three different “return” commands. A return command dictates what, if anything, is returned to the parent code that invokes the function. The final line, 13, utilizes the print command, which simply tells the computer to output everything between the parentheses to the computer in plain text format. Inside the parentheses we have some plain text tokens indicated by quotations, which will be output directly to the console.

Between these two tokens, however, there is an invocation of the function "chickenOrEgg", which the computer recognizes as previously defined. In such situations, the print command is put on hold while the computer executes the contents of the invoked function. Whatever is returned by the "chickenOrEgg" function will be included in the console output between the two string tokens. Everything dictated by these 13 lines of code is formally valid and the program will compile and begin to run successfully (code with invalid structure or logic will fail to compile or crash at runtime). In this particular case, there is an infinite recursion loop present that will keep the program from ever completing the print command. When the computer runs the "chickenOrEgg" function, it is immediately directed to return the results of another invocation, which is the chicken function (chosen arbitrarily, we could have started with the egg function and yielded the same results). It then must leave the "chickenOrEgg" function unresolved as it seeks out the contents of the chicken function, which in turn tells it to return with the results of the egg function. It should be clear to see where the infinite loop comes into play, as the chicken and egg functions serve only to invoke each other. The code in these 13 lines are formal, explicit in their function, and unequivocal, while the English text tokens are unambiguous. A chicken, an egg, and the property of "coming first" have no implied meaning in this scenario and simply refer to their explicit definitions, thus all could easily be substituted for equivalents from a formal language. Because of this, I request that the English tokens be accepted as formal despite English being a very informal language.

Though this code segment is ultimately formal, there is an implicit tongue-in-cheek joke referring to the age-old causality dilemma regarding chickens and eggs. The programmer made a metaphorical representation of this paradox using logically valid (but unsound, the code never resolves) language, which in turn conveys humor to other programmers who read the code and realize the implicit reference. This is merely one example of how to successfully achieve implicature in strict formal language; as Grice points out there are ways to craft implicature in the way that thoughts with formal content are expressed. Sarcasm, humor, and colloquialism are all excellent examples of implicature cases. It is arguable that any given instance of these concepts can be expressed equally in formal language without losing any implicatures.

Within claim (B), languages with the possibility of informal conversation are merely incomplete versions of a larger ethereal formal language. This references the modification of or addition to a language beyond its initial state. Colloquialism is used to convey information and expressions that are not strictly facilitated by the given language due to a lack of sufficient vocabulary or syntax. Typically a result of folkloristic communication, colloquialism is when a new piece of language is formed in order to fulfill unsupported communicative requirements. Often, a colloquial term that reaches a certain level of prevalence will be adopted into newer iterations of the language's official documentation (Merriam-Webster including the word "selfie" with a noted origination circa 2002, for example). This is why humans communicate so differently across generations despite using the same languages; There are significant cultural changes and adaptations over time that provide different use-cases of a language which demand new etymology. If you were to apply Theseus's paradox to a language, you might question whether a language featuring significant revision is even the same language as its earlier iterations.

Since we have established that language is fluid and iterative, it is well within reason to assert that a given language could eventually have specific terms and conjunctions for every possible utterance. We could eliminate all homonyms, pronouns, or anything else that relies on context and implicature in order to accurately and reliably convey information. The resulting language would be sufficient for both formalists and informalists, would it not? I do not believe, however, that it would be a sufficient language for human beings as a whole. We enjoy our implicatures, as it allows for a certain subtlety that is essential for conveying more than raw informational data.

Though I suspect it would certainly be possible to construct a dichotomous key for every human emotion that would allow for succinct and abbreviated expression, I don't think it would be favored over the linguistics of today. Sometimes people prefer to hint at a feeling or idea without explicitly declaring it, like hinting to a friend with codewords or body language that you're bored and want to leave the party. Though the usefulness of formal languages are undeniable and technically boundless, I do not believe they are always preferred or socially appropriate. And though informal language is provably less efficient for logic-oriented tasks, it is not invalidated or deprecated as a means of communication. Formalists and informalists should focus their efforts instead on the validity of implicatures, regardless of the language.