Towards functional, agent-based models of dogwhistle communication

Robert Henderson and Elin McCready

Introduction

[3] build a novel theory of so-called 'dogwhistle' communication by extending the *social meaning games* of [2].

• This work reports on an ongoing project to build systems to model the evolution of dogwhistle communication in a population based on probability monads [4, 5].

• The initial results presented here is a computational implementation of (henderson-mccready), which will serve as the basis for models with multiple speakers and repeated interactions.

Background

Dogwhistling is a communicative act that send messages that only some members of the audience can hear—e.g., George Bush's 2003 State of the Union address, which contains the following line.

(1) Yet there's power—wonder-working power—in the goodness and idealism and faith of the American people.

To most poeple, wonder-working power is at best a civil-religious banality. But, to evangelical Christians, it is phrase from a well-known hymn. To those in the know, this signals Bush is an evangelical.

Selected References

[1] Dylan Bumford and Simon Charlow. 2018. Prob-tools.

[2] Heather Burnett. 2017. Sociolinguistic interaction and identity construction: The view from game-theoretic pragmatics. Linguistics and Philosophy.

[3] Robert Henderson and Elin McCready. 2019. Signaling without Saying: The Semantics and Pragmatics of Dogwhistles. To appear from Oxford University Press.

[4] Martin Erwig and Steve Kollmansberger. 2006. Functional pearls: Probabilistic functional programming in Haskell. Journal of Functional Programming, 16(1):2134.

[5] Eric Kidd. 2007. Build your own probability monads. Draft paper for Hac 07 in Freiburg, 7.

What are probability monads?

The probability monad toolkit as described in [5, 4] is built on a set of monad transformers that enrich monads with probabilistic notions that can be computed in the background (e.g., weights, Bayes' theorem, etc.), separating them from code describing the structure at hand.

• The PerhapsT monad transformer attaches probabilities to each computation in the list monad

• The MaybeT monad transformer allows us to throw out branches of the computation that fail

We can then, for instance, implement of Bayes' theorem via normalizing probabilities of non-failed branches of a compution.

Sociolinguistic Signaling Games with probability monads

We can treat priors over messages as probability distributions over lists, a fundamental concept in this programming paradigm.

```
messagePrior :: Dist m => Group -> Persona -> m Message
messagePrior Ingroup [p1,p2] = weighted [Mass 80 m1,...
messagePrior Naive [p1,p2] = weighted [Mass 15 m1,...
```

Then, following [1]'s previous work on simply RSA in probability monads, we have a super clean definition of utility for social meaning games.

```
-- How different groups of speakers and listeners value various personas
vL :: Group -> Persona -> Float
vS :: Persona -> Float

-- Social utility calculation
uSoc :: Message -> Persona -> Group -> Lexicon -> Float
uSoc m p g l = pr + (vL g p * pr) + (vS p * pr)
  where Sum pr = sum $ [x | Mass x (Just y) <- runMassT (runMaybeT
  (listener 1 g m eval)), y == p]
```

Why is this good?

• It is good to be able to confirm that your theories work. Our paper describes a computational implementation of a chapter of our book. It type checks and produces the dogwhistle effect.

• Probability monads allow us to completely separate our theory and the probabilistic programming underlying the computations we have to perform to explore its outputs on specific examples.

• We now have a base for considering populations of agents playing these games. This will require an even higher level of abstraction, but it will be easier to implement because, once again, we can ignore all the bookeeping, which is handled in the background by the monad transformers.

The Code

https://github.com/bkeej/SocialMeaningExp/blob/master/src/RSAsoc.hs

H & McC on Dogwhistles

Literal listener computes the probabilty the speaker the speaker bears a persona given the received message:

$$L_0(p|m) \propto P(m|p) \times P(p)$$

A sociolinguistically aware speaker picks a message to maximize their payoff given it will be received by the literal listener—$U_{S_1}(m, L_0)$—accounting for both of their preferences for various personas:

$$\sum_{p \in [m]} P_{L_0}(p|m) + \nu_{S_1}(p) P_{L_0}(p|m) + \nu_{L_0}(p) P_{L_0}(p|m)$$

Why agent-based models?

Dogwhistles should evolve under the following conditions:

• Ingroup members, in virtue of speaking with each other, should develop linguistic variants that occur at a higher rate than outgroup members,

• Most outgroup members (naive) should be unaware of these linguistic variants that signal ingroup members, though some savvy outgroup members may be away of ingroup language, and,

• Group membership is punished by outgroup members, but rewarded by ingroup members.

In order to see if this hypothesis is correct, we would like to model groups of agents interatively playing sociolinguistic signaling games of the kind discussed, over a structured population—i.e., some subgroups of speakers more likely to interact.

What next?

Easy:

• Add populations of agents—i.e., updateable priors for how well bits of language express personas and preferences over personas.

Harder (at least to theorize about):

• A route towards innovation, so either a way for novel expressions to enter the games or for a rich enough stable of expressions for ingroup / outgroup variation to evolve.