Non-parametric Bayesian Methods in Machine Learning

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 11, 2014

Outline

- (My) Bayesian philosophy
- Gaussian Processes for Regression and Classification (Monday)
  - GP preliminaries
  - *Application 1*: typing on touch-screens
  - Classification (including semi-supervised)
  - *Application 2*: clinical (dis)-agreement
- Dirichlet Process flavoured Cluster Models (Tuesday)
  - DP preliminaries
  - *Application 3*:Idenfitying metabolites
  - *Application 4*:Cluster models for multiple data views
- Summary

Relevant publications

- The four applications are described in the following papers:
  - Uncertain Text Entry on Mobile Devices Weir et. al, CHI 2014
  - Investigating the Disagreement Between Clinicians' Ratings of Patients in ICUs Rogers et. al 2013, IEEE Trans Biomed Health Inform
  - MetAssign: Probabilistic annotation of metabolites from LC–MS data using a Bayesian clustering approach Daly et. al, Bioinformatics, under review
  - Infinite factorization of multiple non-parametric views Rogers et. al, Machine Learning 2009

About me

- I'm not a statistican by training (don't ask me to prove anything!).
- Education:
  - Undergraduate Degree: Electrical and Electronic Engineering (Bristol)
  - PhD: Machine Learning Techniques for Microarray Analysis (Bristol)
- Currently:
  - Lecturer: Computing Science
  - Research Interests: Machine Learning and Applied Statistics in Computational Biology and Human-Computer Interaction (HCI)

Lecture 3: Application: Touchscreen typing

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 11, 2014

Typing on touchscreens

- Most people have smartphones
- Most smartphones have touchscreens
- Touchscreens are small
- Keyboards on touchscreens are small
- Typing on them is hard!
    - ... but people type on them a lot

Background 1: Why is it hard?

- Occlusion of target by finger
- ‘fat finger’ problem
- Small targets
- Demo: `http://bit.ly/1nBws97`

Background 1: Why is it hard?

- Occlusion of target by finger
- ‘fat finger’ problem
- Small targets
- Demo: http://bit.ly/1nBws97
- Quite a bit of work in this area:
  - Holz and Baudisch
  - Henze (100,000,000 taps)
- Collecting data is fairly easy

Background 2: All users are different

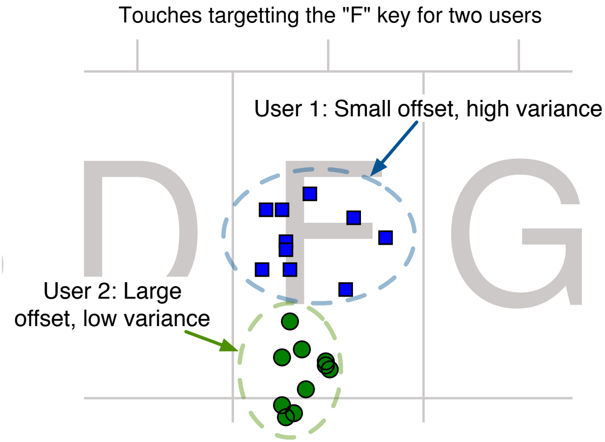


Figure 5 : Touches recorded by two users aiming for the 'F' key. User 2 has high bias and low variance, user 1 has low bias and high variance.

Background 3: Current systems (maybe?)

- Touch is boxed into nearest key.
- Key ID is passed to a Statistical Language Model (SLM).
- SLM is made up of probabilities of observing certain character strings (from large text corpora).
- SLM can swap characters to make the character string more likely.
  - e.g. ‘HELLP $\rightarrow$ HELLO’

Our idea

- There is a lot of uncertainty present in touch (bias and variance)
- Boxing a touch into a key is probably bad
- Why can't we pass a *distribution* to the SLM?
  - Pass the uncertainty onwards
  - Being Bayesian!

Our idea

- There is a lot of uncertainty present in touch (bias and variance)
- Boxing a touch into a key is probably bad
- Why can't we pass a *distribution* to the SLM?
  - Pass the uncertainty onwards
  - Being Bayesian!
- Can use a user specific GP regression model to predict target from input touch.

Our idea

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

Our idea

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

Our idea

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

Our idea

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

Our idea

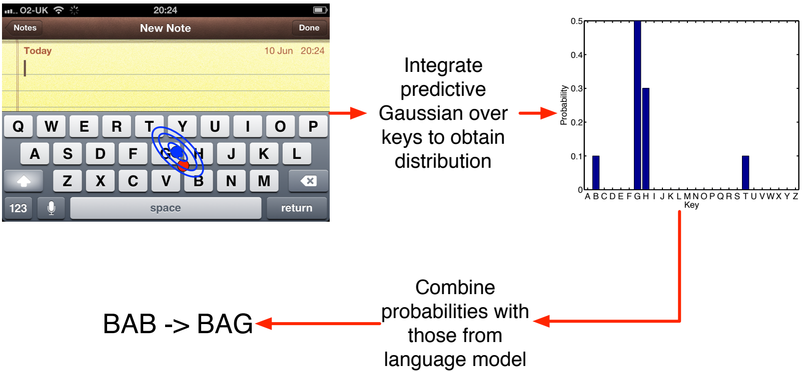


Figure 7 : The complete system

The model

- We use independent GP regressions for predicting x and y offsets.
- Training data:
  - Each user typed phrases provided to them.
  - Data: the x, y location of the recorded touch. Target: the center of the intended key minus the touch (i.e. the offset).

The model

- We use independent GP regressions for predicting x and y offsets.
- Training data:
  - Each user typed phrases provided to them.
  - Data: the x, y location of the recorded touch. Target: the center of the intended key minus the touch (i.e. the offset).
- Used a GP with zero mean and a composite covariance:

$$C(\mathbf{x}_1, \mathbf{x}_2) = a\mathbf{x}_1^T\mathbf{x}_2 + (1-a)\exp\{-\gamma||\mathbf{x}_1 - \mathbf{x}_2||^2\}$$

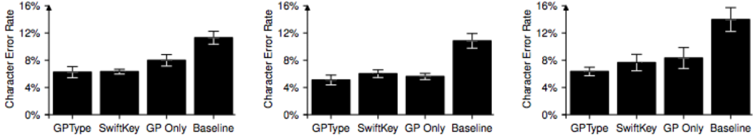
Video

- http://www.youtube.com/watch?v=llQI5gV5l74

The experiment

- 10 participants
- Calibration data collected for each
  - Note: calibration task matters
- each did 3$\times$ 45 minute sessions, typing whilst sitting, standing and walking. [more details in paper]
- Compared:
  - GPtype (our system), Swiftkey (commercial Android keyboard), GP only (just offset, no SLM), baseline (boxing, no SLM).

Results



(a) Sitting (b) Standing (c) Walking

Figure 4. Character error rates for the two keyboards we evaluated, separated by mobility condition (Study 2). Plots show mean and standard error across all participants. The baseline method represents the literal keys touched, while GP Only shows the keys hit after the mean GP offset is applied.

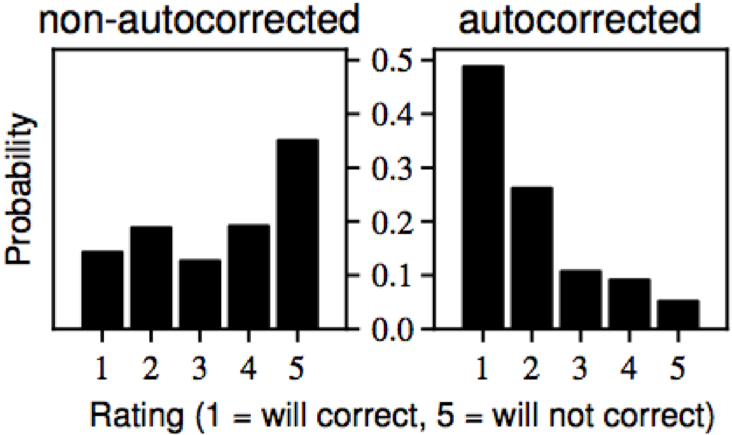
Figure 8 : Results of GPType experiment

- GPType marginally (stat sig) better than Swiftkey.
  - A **lot** of people work on SwiftKey
- Baseline awful!

Explicit uncertainity control

- In GPType, uncertainity is handled implicitly
- As user typing becomes more uncertain, more power given to language model
- Could users *explicitly* control this?
  - Certain inputs: no SLM control (slang, names, etc)
  - Uncertain inputs: high SLM control
- Use pressure to control certainty:
  - High pressure: high certainty
  - Low pressure: low certainty

Do users know when SLM will fail?



- Users given phrases and asked whether they thought autocorrect would change them incorrectly
- Users quite good at understanding SLM failings

ForceType

- Modified Synaptics Forcepad
- Pressure mapped to Gaussian variance (no GP)
- System explained to users
- Users type phrases with and without forcetype

ForceType: Results

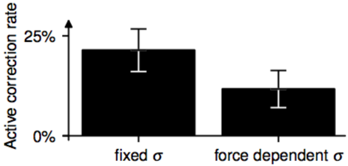


Figure 8. ForceType requires significantly fewer active corrections from users when entering text. Required corrections dropped by ≈ 10 percentage points. Errors bars are 1 sd.

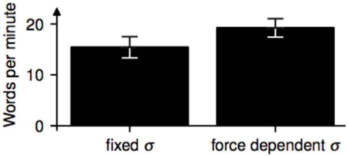


Figure 9. ForceType enabled users to enter phrases $> 20\%$ faster. A significant increase over the baseline. Errors bars are 1 sd.

- Forcetype reduced number of corrections performed by users (top)
- Forcetype improved overall text entry rate

Conclusions

- GP regression is key to the approach: we make no parametric assumptions (what would they be?)
- . . . and get probabilistic predictions
- . . . that can be fed to the SLM – (un)certainity is passed to the SLM
- Performance is promising

Conclusions

- GP regression is key to the approach: we make no parametric assumptions (what would they be?)
- ... and get probabilistic predictions
- ... that can be fed to the SLM – (un)certainity is passed to the SLM
- Performance is promising
- Can also use pressure to provide explicit uncertainty control

Conclusions

- GP regression is key to the approach: we make no parametric assumptions (what would they be?)
- ... and get probabilistic predictions
- ... that can be fed to the SLM – (un)certainity is passed to the SLM
- Performance is promising
- Can also use pressure to provide explicit uncertainty control
- More info:
  - http://www.youtube.com/watch?v=l1QI5gV5174
  - http://pokristensson.com/pubs/WeirEtAlCHI2014.pdf
  - Acknowledgements: Daryl Weir, Per Ola Kristensson, Keith Vertanen, Henning Pohl