Smart Recipe Measurements with Learned Volume Prediction

Kurt Nelson
Sam Maticka
{knelson3, smaticka}@stanford.edu

Overview

Would you rather chew off your pinky or measure ingredients for the rest of your life? The response is unanimous - no one needs a pinky anyhow. No one likes measuring, but previous research (Nelson&Maticka,2017) suggests that the Italian Grandma Method (IGM) – a splash of this and a dab of that - produces gastronomical grenades for all but the most experienced chefs (e.g. IG's).

Many of us casual cooks are not able to reproduce winning recipes when applying the IGM, and instead are left with a one-hit-wonder followed by an onslaught of failed recreation attempts that pale in comparison and litter your fridge with dreadful leftovers. We propose a solution to this dystopia.

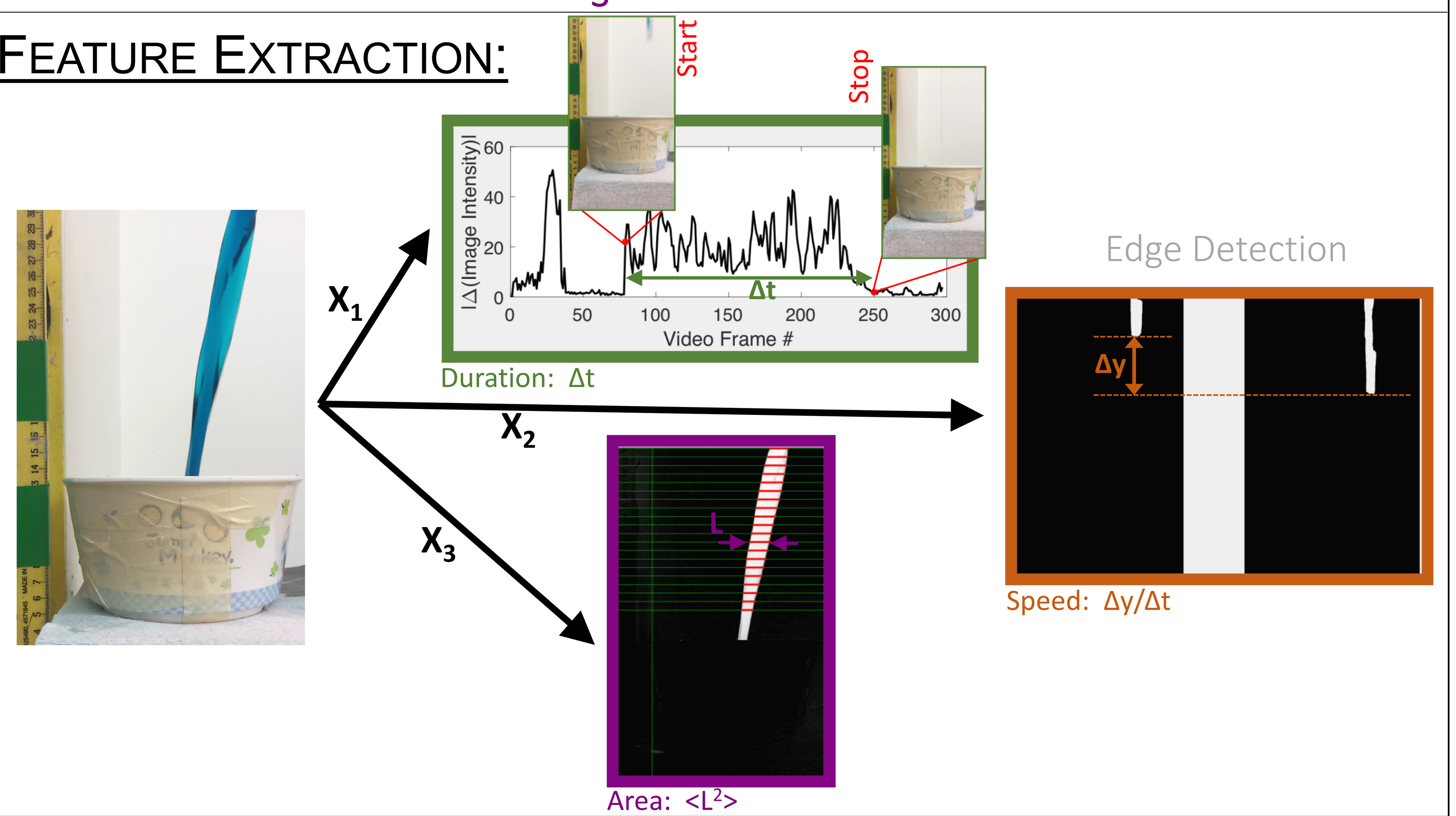
The broader purpose of this project is to create a smart recipe recorder and instructor. You can either 1) make a recipe-free dish, and add ingredients at will while a smart device films and records the recipe, or 2) you can create a saved recipe, and have the device tell you when to stop pouring a specified ingredient.

We applied machine learning and computer vision to teach phones how to measure ingredients for us. We chose to focus on poured liquids as a first step.

Data Collection

Pouring Videos → (Feature Extraction) → X_1: Duration, X_2: Speed, X_3: Area → (Machine Learning) → Predicted Volumes

Feature Extraction:



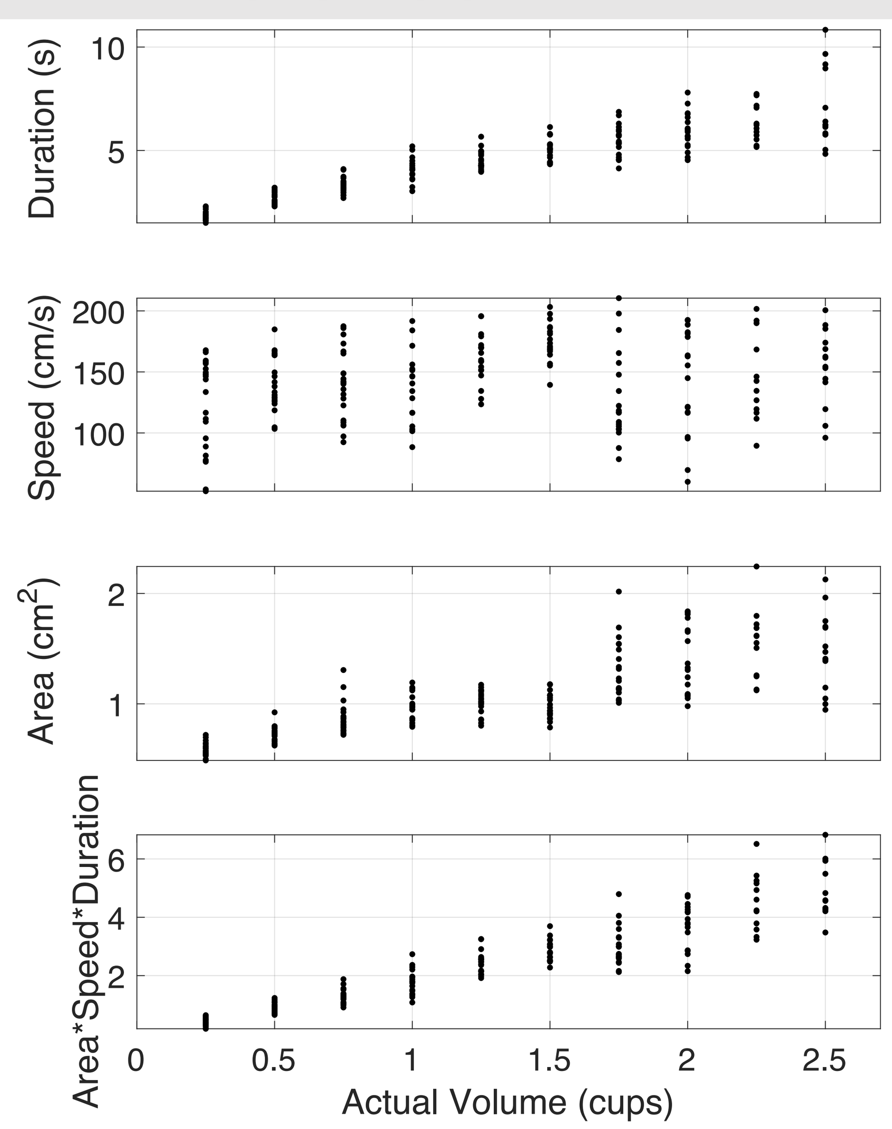
Features & Selection Process

Feature Properties:

- 3 fundamental physically-relevant features
- Inherent feature variance
- 3-way interaction term is a physical estimate of volume

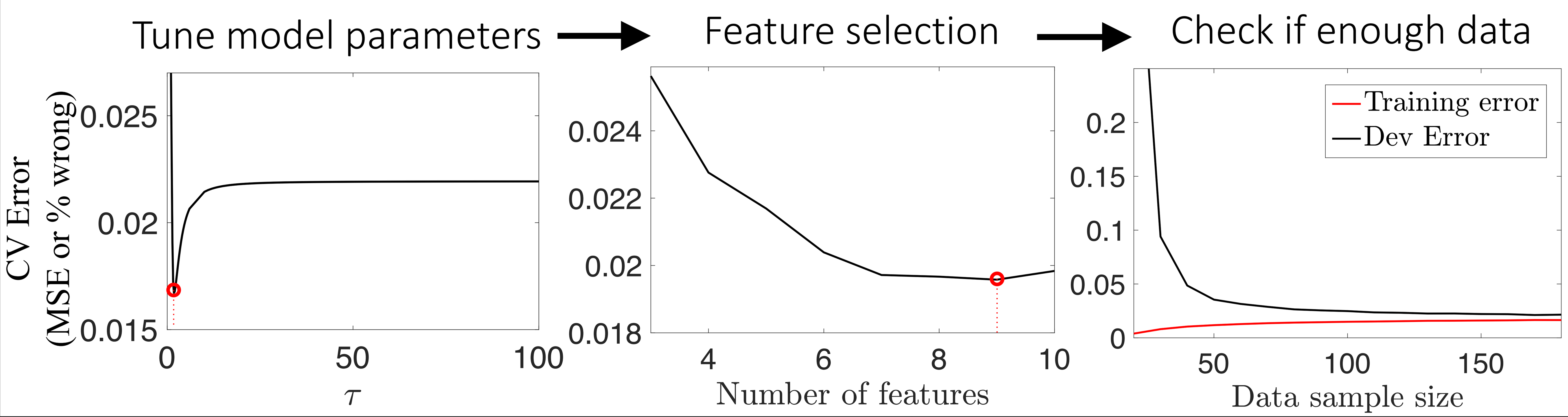
Feature Selection Process:

- Forward feature selection for each model type
- Full feature set: 3 fundamental features, 2nd-order terms, 2-way interactions, and 1 3-way interaction (see Feature Properties)
- CV error with k=10 folds (split: 90/10) was used as the selection metric
- The feature set that yielded the lowest CV error was chosen for the respective model



Model Selection/Assessment Method

Split Data: 90/10



Select Model With Lowest Development Error

Test on Reserved Data (10%)

Model Testing

Development Set Errors	
Regression Model	MSE (cups)
1) Weighted Least Squares	$\boxed{0.017}$
2) K-Nearest Neighbors	0.018
3) Ordinary Least Squares	0.021
4) Ridge Regression	0.021
5) Lasso Regression	0.023
Classification Model	Misclassification Error (%)
1) Softmax	$\boxed{25}$
2) K-Nearest Neighbors	26
3) Linear Discriminant Analysis	28
4) Support Vector Machines	42
5) Physical Model (rounded)	78

Chosen Models

Locally Weighted Linear Regression

- $h(x^m) = \sum_{j=1}^{n} \theta_j x_j^m$
- $J(\theta) = \sum_{i=1}^{m} w^i (h(x^i) - x^i)^2$
- $w^i = \exp[-\frac{(x^i - x)^T (x^i - x)}{2\tau^2}]$, Tuned τ=2.2
- Selected 7 features: 3 fundamental features, 3-way interaction, time^2, $(duration)(area)$, $(speed)(area)$
- MSE error on unseen test data was 0.016 cups

Physical Model

- $h(x^i) = \alpha x_1^i x_2^i x_3^i$ $=\alpha(area)(duration)(speed)$
- $\alpha = \sum_{i=1}^{m} \frac{y^i}{x_1^i x_2^i x_3^i}$

Softmax

- $p(y = j | x; \theta) = \frac{\exp(\theta_i^T x)}{\sum_{j=1}^{k} \exp(\theta_j^T x)}$
- Maximize: $\ell(\theta) = \sum_{i=1}^{m} log \prod_{l=1}^{k} \left(\frac{\exp(\theta_l^T x)}{\sum_{j=1}^{k} \exp(\theta_j^T x)} \right)^{1\{y^i=1\}}$
- Selected 6 features: 3 fundamental features, 3-way interaction, $duration^2$, $speed^2$
- Misclassification error on unseen test data was 25%

Conclusions

- Machine learning does far better than a baseline prediction using theory:
  - MSE on test data: 0.016 cups vs. 0.072 cups
  - Misclassification Error: 25% vs. 75%
- Non-parametric models perform better than parametric
- Data sample size was adequate for simple regression models tested. This was confirmed by convergence of test and development errors. However, the data sample size limited the classification models we were able to test.

Future Work

Test Other Models:
- Neural networks. We would need to collect a lot more data
- PCA combined with other methods
- Regression Trees (preliminary results show promise)

Things that may help current models:
- Improve experimental setup - stereo cameras for better cross-sectional area estimate
- Improve feature extraction algorithm:
  - Better estimate of flow rate (speed) - some sort of intermittent particle tracking
  - More rigorous removal of erroneous cross-sectional area

Expansion of the smart device's abilities:
- Generalize the model for different ingredients – dry ingredients, clear liquids,