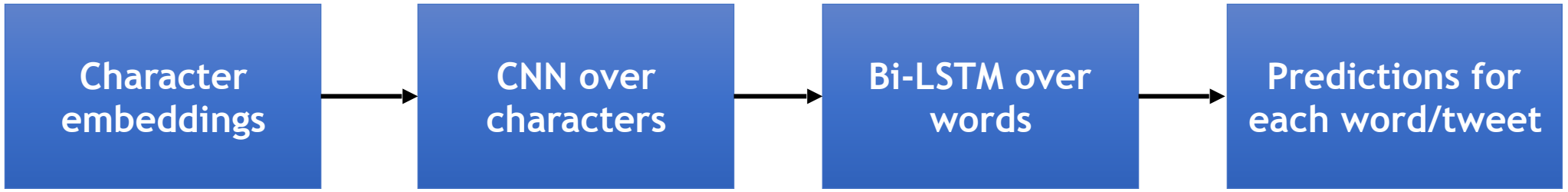
A Neural Model for Language Identification in Code-Switched Tweets

Aaron Jaech, George Mulcaire, Shobhit Hathi,
Mari Ostendorf, Noah A. Smith

UWNLP

This talk in a nutshell

- Continuous-space (neural) representations
- Hierarchical approach for multiple kinds of context
  - Char n-grams should be interpreted in the context of a word
  - Words should be interpreted in the context of their neighbors
- Tweet-level langid for similar languages and many languages
- Good performance for Spanish/English code-switching



Standard language ID approaches

- Classifiers over character n-grams and word dictionary features
- Character n-grams are highly informative… but limited

Rodri
@RodrigoGomez13

Quien entiende nuestro clima?

CarlosZetina
@VosdecimeCarlos

Qieen entiende a los adultos..

- Want to reason about a broader notion of similarity

Continuous representations for language

- Continuous space representations (neural networks) have led to progress in language modeling, machine translation, and other tasks
- Key advantages vs n-gram models:
  - Share information over longer time ranges (recurrent neural net)
  - Soft parameter tying
- Our main contribution: use the advantages of continuous space representations in the task of language identification

Why is a CNN like an n-gram model…

- Each filter in a convolutional neural network sees a fixed-width character sequence
- Filters act like linear classifiers indicating the presence or absence of a generalized n-gram
- Max-pooling creates fixed size representation for each word

…but different?

- Sharing information between similar characters and patterns
- Incorporating wider context into a single feature vector

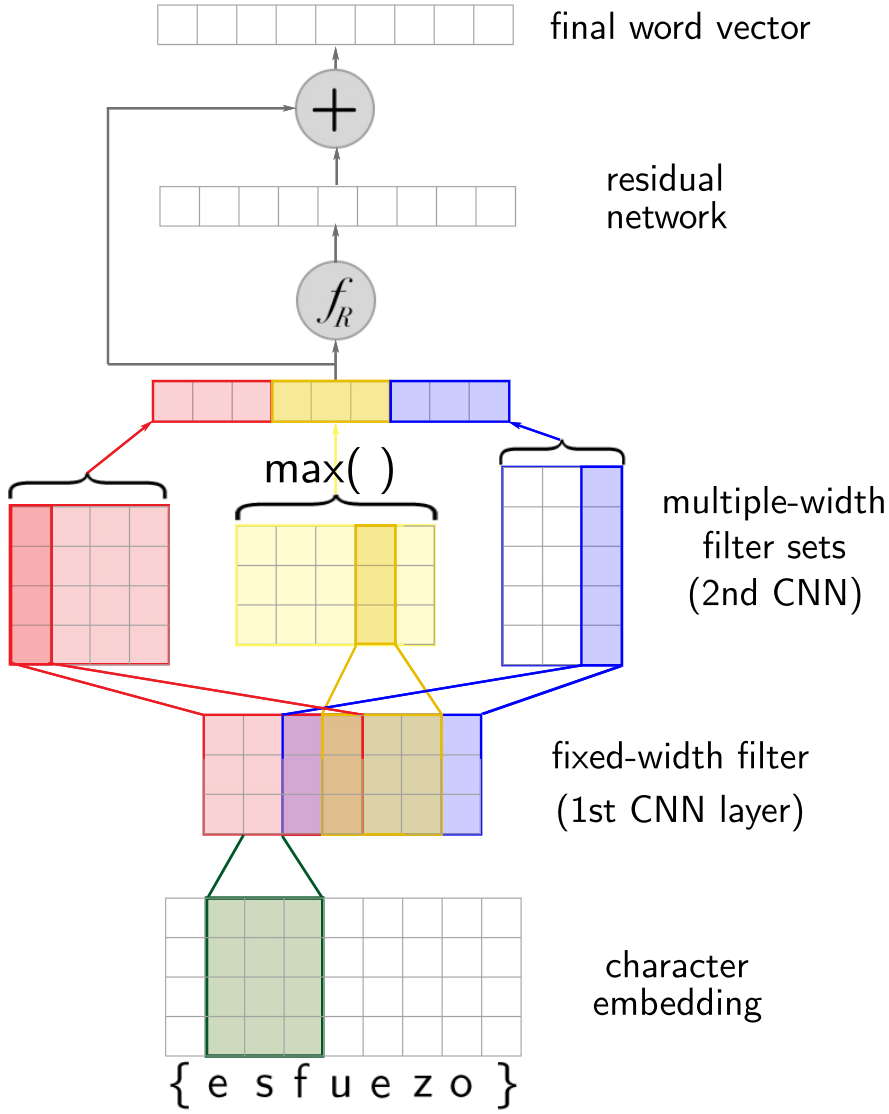
Convolutional network example

Input Sequence:	<S>	Y	O	U	R	S	</S>	Pooled Vector
Filter 1 Output:	X	X	✓	X	X	X	X	✓
Filter 2 Output:	X	X	X	X	X	✓	X	✓
Filter 3 Output:	X	X	X	X	X	X	X	X

- Each filter sees three characters at a time (WIDTH = 3)
- Hypothetical example:
  - Filter 1 detects “YOU” trigram
  - Filter 2 detects words ending in ”S”
  - Filter 3 detects “ING” trigram
- Pooled vector indicates presence of pattern in the input word

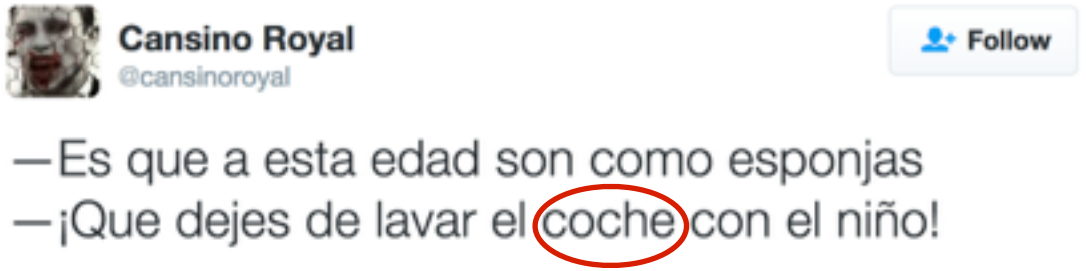
Char2Vec

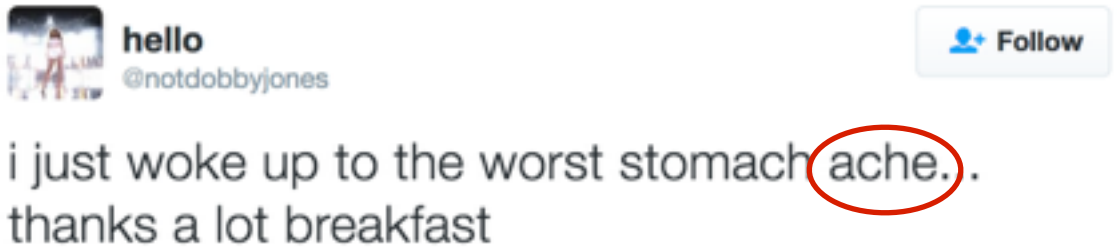
- Embeds each space-delimited word
- Multiple width filters (from 3 to 6) capture different length patterns
- Residual network allows feature interactions
- RELU used as an activation function



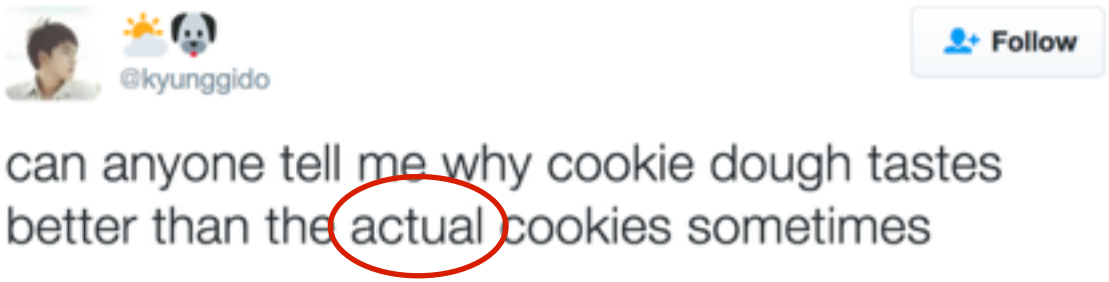
Are word representations enough? (no)

- Similar patterns can be misleading





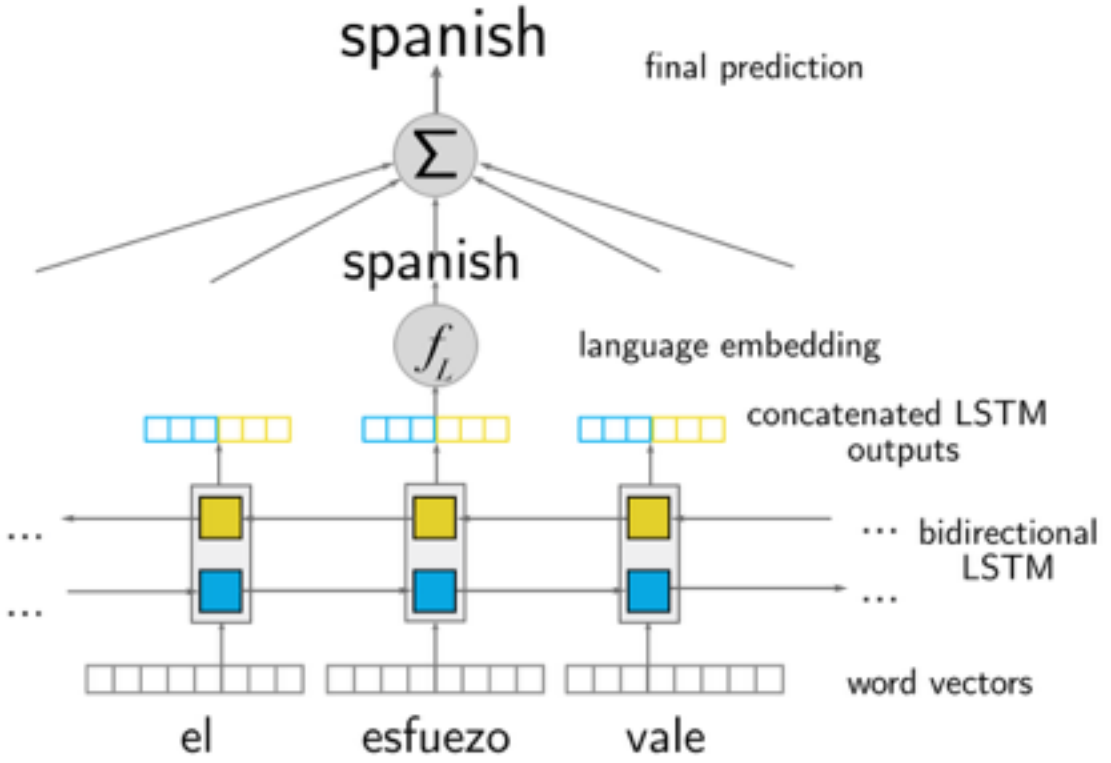
- Exact matches across languages



- Placing words in context is nontrivial:
  - "que dolor, actual worst headache"

Tweet level context

- Take as input the sequence of word vectors from Char2Vec
- Process with a bi-directional LSTM
  - LSTM input gates minimize the effect of URLs and usernames
- Make predictions for each word
- Final prediction is average of words' labels



- We call this method C2V2L (character to vector to language)

Development datasets

Two hand-labeled Twitter datasets:

- Twitter70 covers 70 languages from five language families
- TweetLID is a small set of confusable Iberian languages plus English with some ambiguous Tweets and some code-switching
- Tweet-level supervision

	Twitter70	TweetLID
Tweets	58,182	14,991
Character vocab	5,796	956
Languages	70	6
Code-switching?	Not Labeled	Yes
Balanced?	Roughly	No

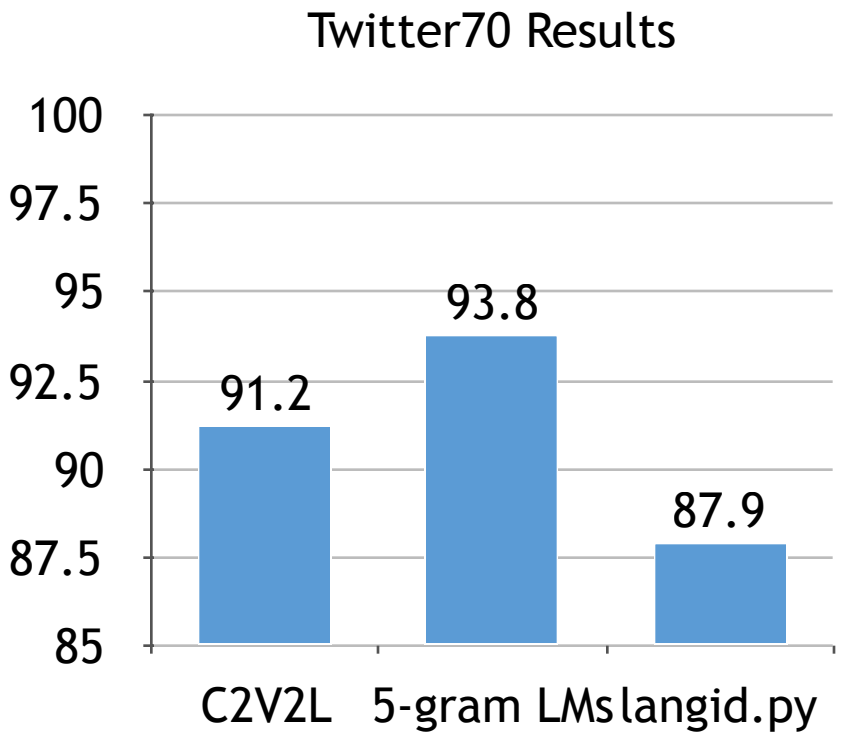
Baseline models

- N-gram language models:
  - Train a separate character 5-gram language model for each language
  - Each LM estimates P(TEXT|LANGUAGE)
  - Pick the label that has the highest posterior probability
    - argmax P(TEXT|LANGUAGE) P(LANGUAGE)
  - Special threshold used for the unknown language label
- Convolutional Model Baseline (C2L):
  - Treat entire tweet as a single character sequence
- Published baseline: langid.py (Lui and Baldwin, 2012)
  - Popular open source classifier that uses character n-gram features

Twitter70 results

Training Examples	Languages
0-200	Oriya
200-400	Uighur, Lithuanian
400-600	Danish, Amharic, Tibetan, Korean, Basque, Chinese, Ukrainian, Tagalog
600-800	Romanian, Turkish, Marathi, Catalan, Taiwanese, Haitian, Icelandic, Malayalam, Serbian, Sinhala, Kannada, Kurdish, Urdu, Hungarian, Punjabi, Cambodian, Latvian, Lao
800-1000	Polish, Japanese, Hindi-Latin, Finnish, Tamil, Burmese, Dutch, English, Bengali, Russian, Georgian, Portuguese, Pashto, Vietnamese, Gujarati, Slovak, Italian, Telugu, French, Estonian, Divehi
1000-1200	Persian, Nepali, Greek, German, Arabic, Bulgarian, Swedish, Armenian, Norwegian, Czech, Spanish, Slovene, Indonesian, Bosnian, Welsh, Thai, Croatian
1200+	Hebrew, Hindi, Sindhi

- Limited by performance on most difficult language pairs
- Croatian/Bosnian, Danish/Norwegian



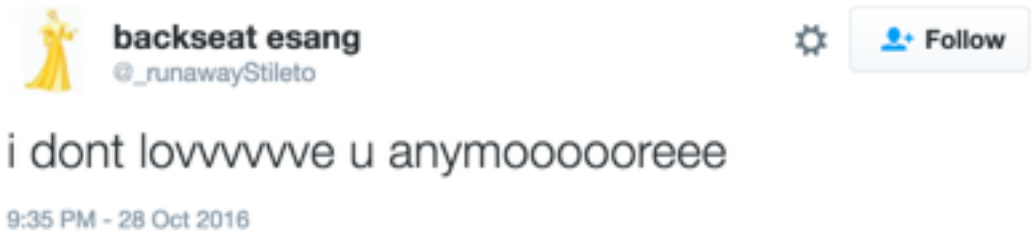
TweetLID results

- Our model beats N-gram baseline and langid.py
- Previous state of the art on this dataset is 75.3 F1 (Gamallo et al., 2014)
- Hierarchical model is substantially better than non-hierarchical neural network (C2L)

Model	Avg. F1	eng	spa	cat	eus	por	glg	und	amb
N-gram LM	75.0	74.8	94.2	82.7	74.8	**93.4**	49.5	**38.9**	87.0
Langid.py	68.9	65.9	92.0	72.9	70.6	89.8	52.7	18.8	83.8
C2L	72.7	73.0	93.8	82.6	75.7	89.4	57.0	18.0	92.1
C2V2L	**76.2**	**75.6**	**94.7**	**85.3**	**82.7**	91.0	**58.5**	27.2	**94.5**

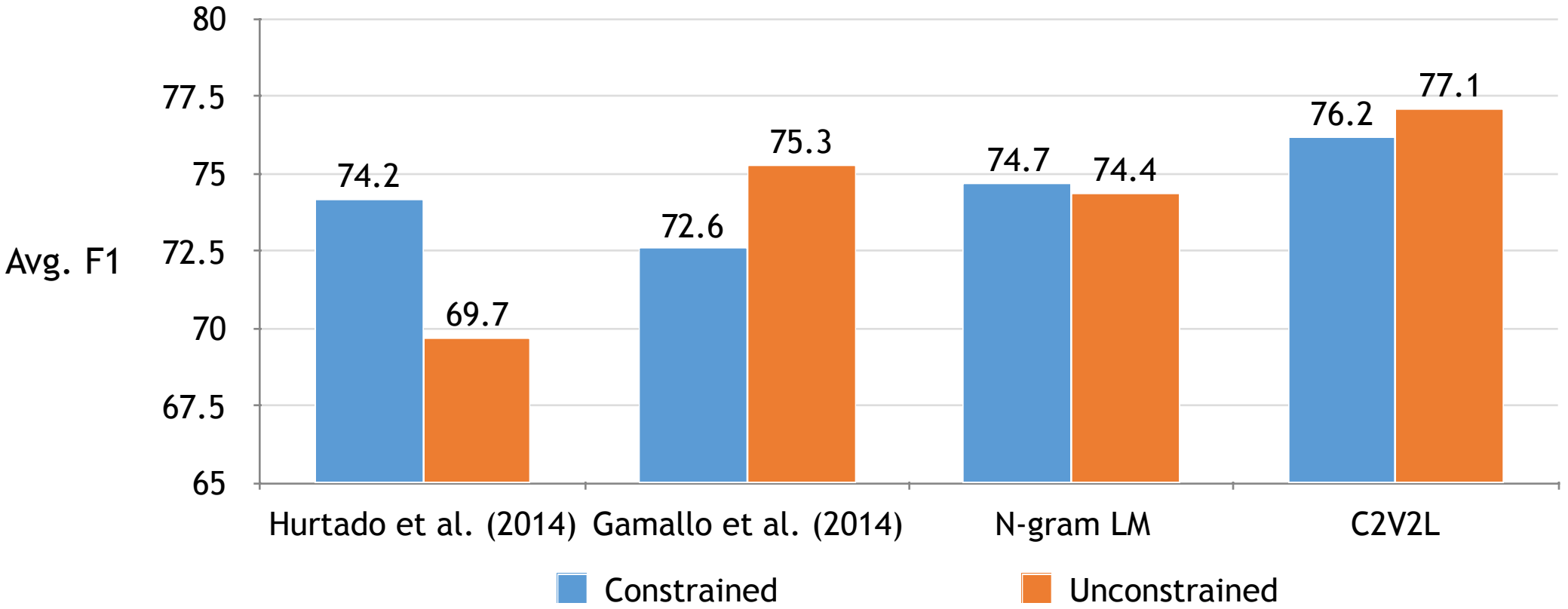
Adding outside data

- Surprisingly, most participants in TweetLID workshop did worse in unconstrained track
- 25,000 Wikipedia sentence fragments per language
- Train model on weighted mixture of wiki and tweets
- Use it to initialize a second model that is fine tuned on just tweets
- N-gram baseline is interpolated between wiki LM and Twitter LM



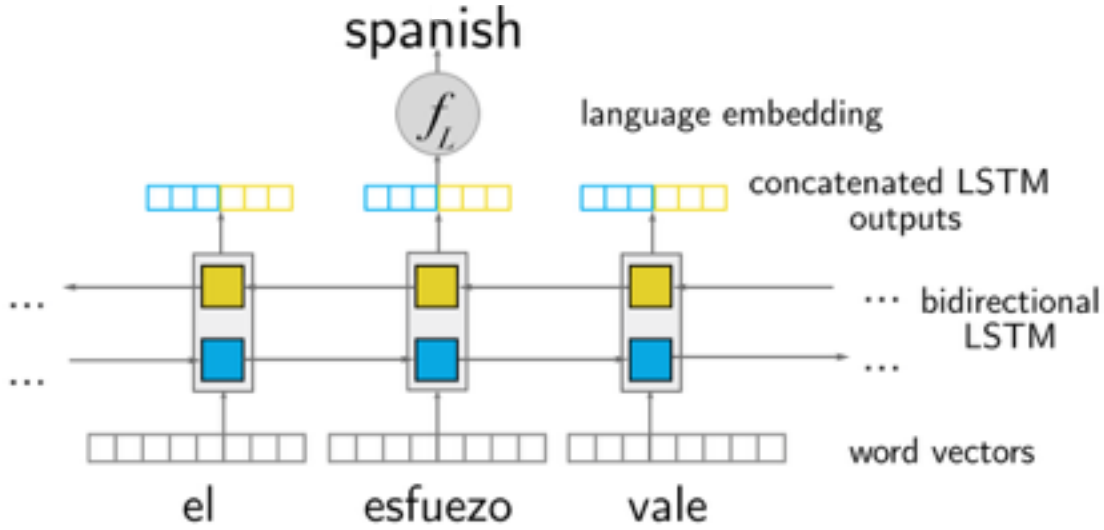
Added data results

- C2V2L beats previous record (Gamallo et al., 2014) for using outside data
- Ability to leverage wiki data is an advantage of our C2V2L model



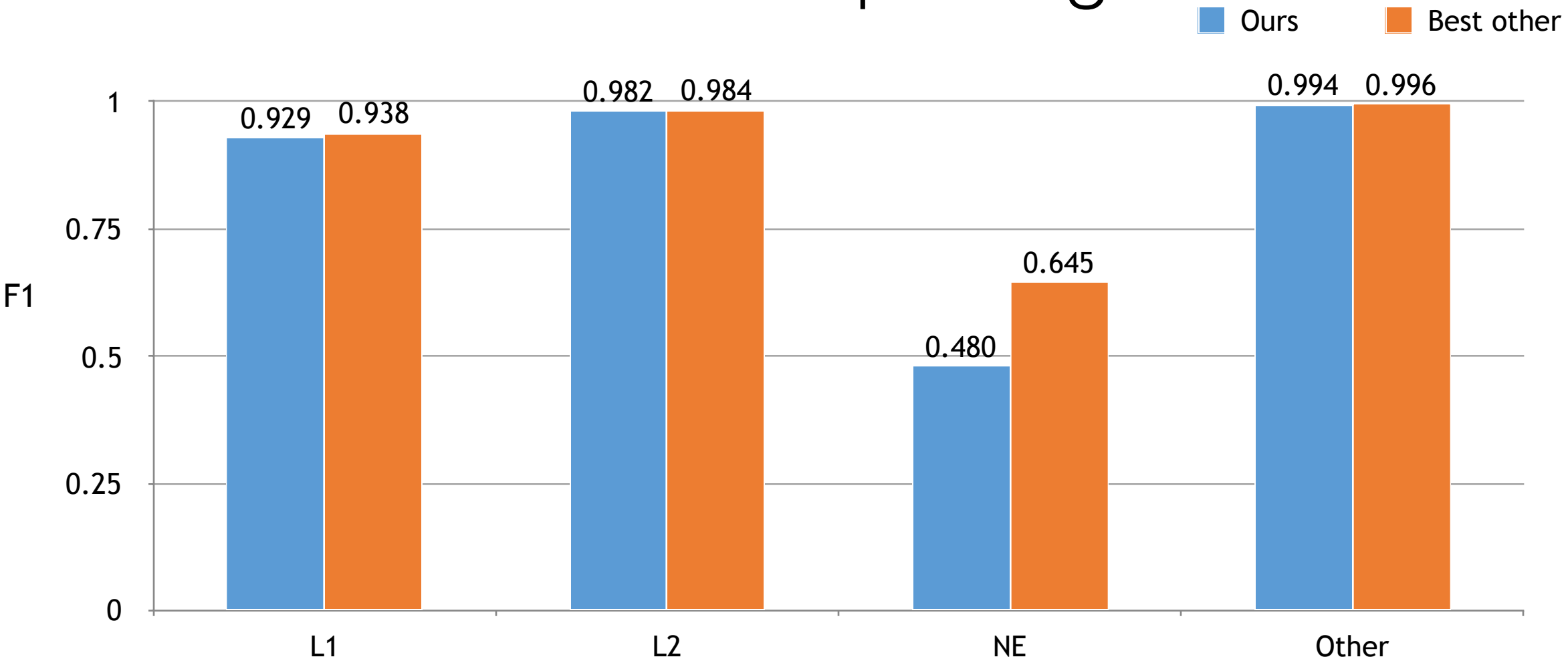
Code-switching

- Our model is easily adapted to predict code-switching
- Just remove the last layer (tweet level prediction averaging)

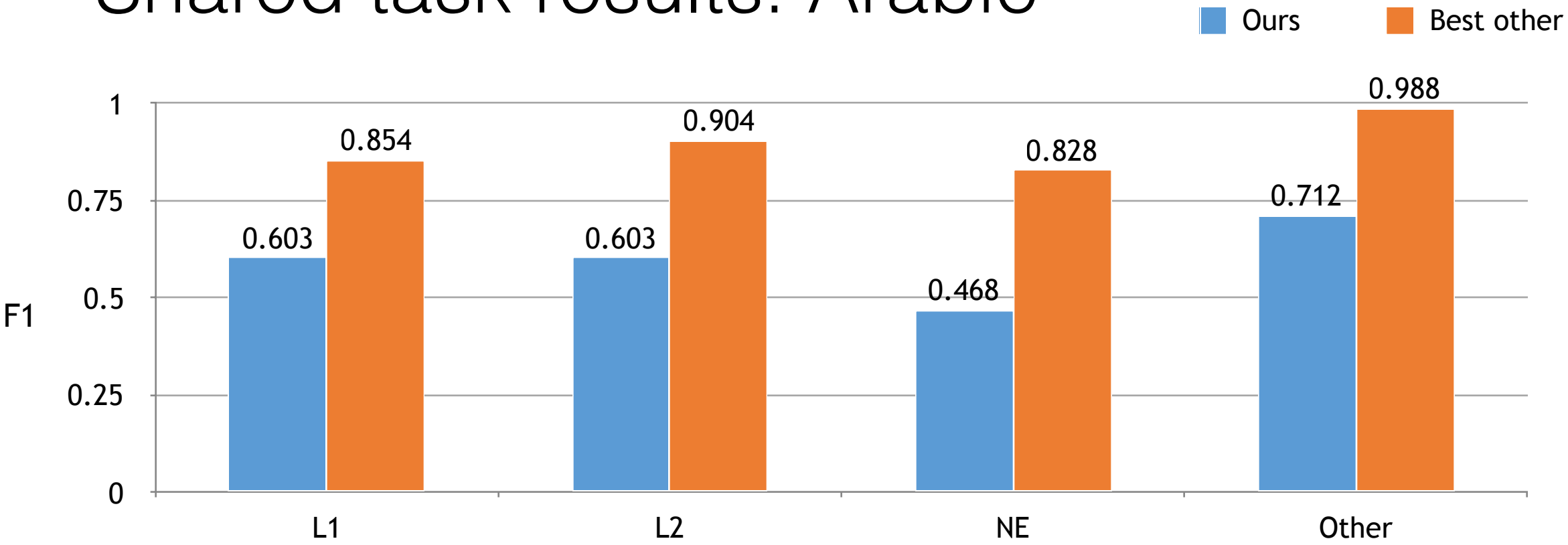


Input:	Vamos	a	echar	un	partido	de	fifa	contra	my	brother	☺️
Predictions:	spa	spa	spa	spa	spa	spa	spa	spa	eng	eng	other

Shared task results: Spa-Eng

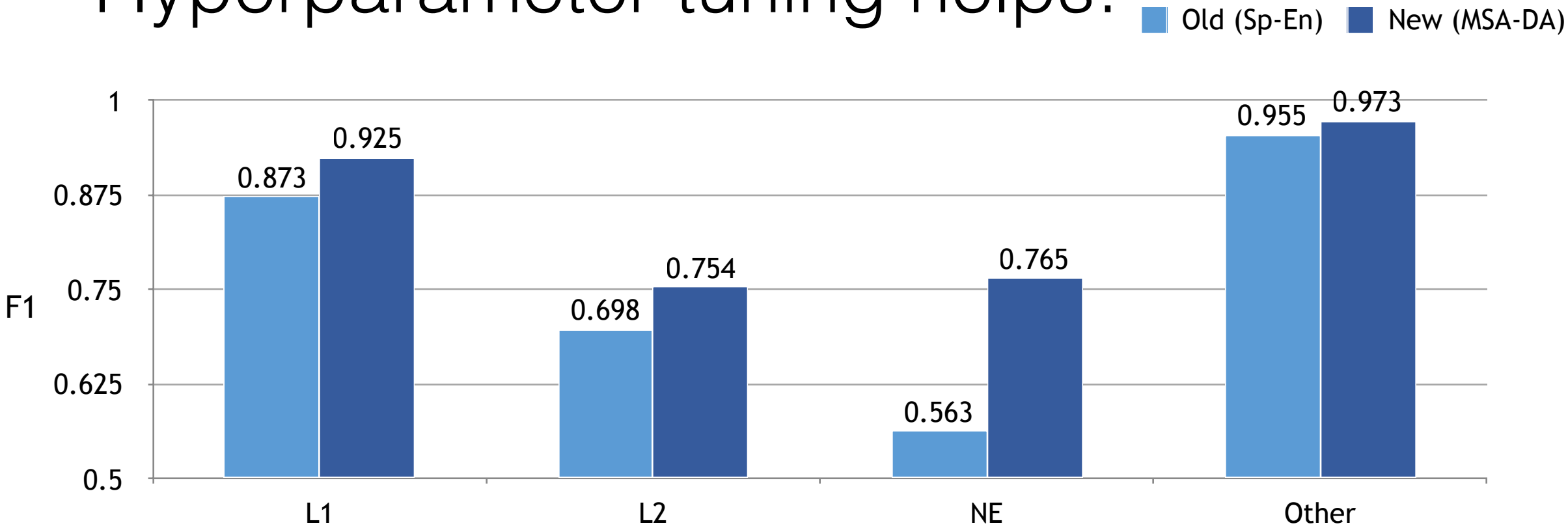


Shared task results: Arabic



- Poor results for Modern Standard Arabic / Dialectical Arabic — but based only on Spanish-English hyperparameters

Hyperparameter tuning helps!



- Arabic-specific hyperparameters show improvement over the Spanish-English hyperparameters (in dev set comparison)

Error analysis — examples

- “Oversharing” between words

- Named entities

Error analysis — examples

- Ambiguous and unknown words

- Confusing Spanish and English

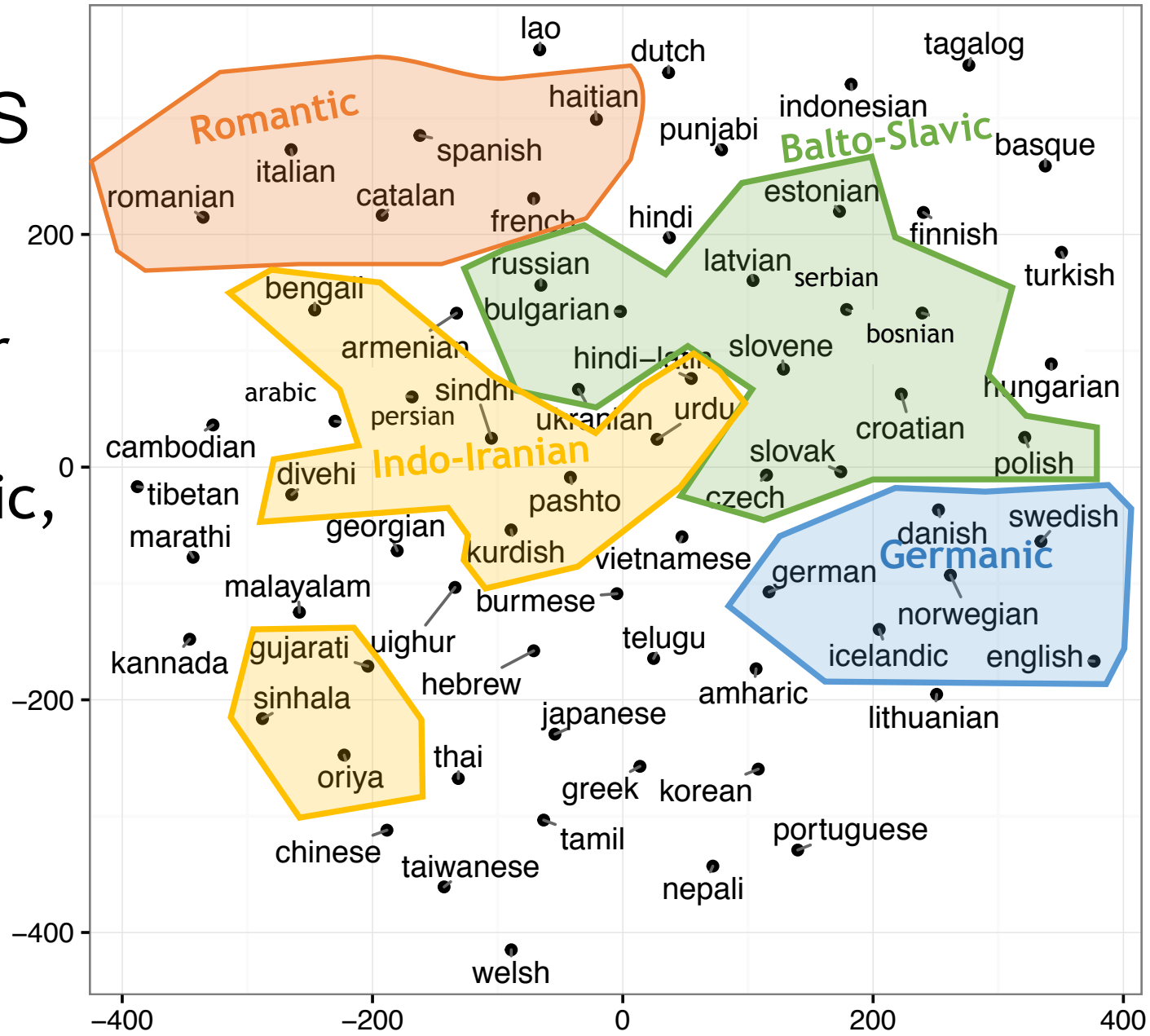
Similar words

- Char2Vec can deal with out of vocabulary words
- Learns to ignore punctuation and capitalization
- Handles non-standard tokens like usernames and hashtags

couldn't		@maria_sanchez		noite	
can’t	0.84	@Ainhooa_Sanchez	0.85	Noite	0.99
don’t	0.80	@Ronal2Sanchez	0.71	noite.	0.98
ain’t	0.80	@maria_lsantos	0.68	noite?	0.98
don’t	0.79	@jordi_sanchez	0.66	noite..	0.96
didn’t	0.79	@marialouca?	0.66	noite,	0.95
Can’t	0.78	@mariona_g9	0.65	noitee	0.92
first	0.77	@mario_casas_	0.65	noiteee	0.90

Language vectors

- Related languages often appear next to each other in the T-SNE embedding
- Orthographic, not phonetic, similarity



Conclusions

- Neural/continuous models have a few advantages including ability to leverage outside data, integration with later stages
- Hierarchical representation is key
- Performance of the model should continue to improve as more training data becomes available
- Simple approach

Future directions

- Joint segmentation and language ID
- Supplement code-switching data with tweet-level supervision
- Other tagging tasks (with joint language ID?)
- Additional features: lexicons, POS, etc.

Links:

- TweetLID: http://komunitatea.elhuyar.eus/tweetlid/
- “Twitter70”: https://blog.twitter.com/2015/evaluating-language-identification-performance (“recall oriented”)
- Our paper on ArXiv: https://arxiv.org/abs/1608.03030
- Contact: gmulc@cs.washington.edu

Thank you!