Natural Language Processing (Almost) from Scratch

Collobert et al., 2011

presentation by **Jack Morris**
12/1/19

https://qdata.github.io/deep2Read/

1.1 - Background

Current NLP representation:

1. Task-specific representations
2. Simplified, intermediate representation based on
   a. Syntactic information
   b. Semantic information
3. Benchmark corpora, manually annotated, to evaluate performance

Drawback:

1. Representation not generally applicable across different corpora
2. Reveals little about how to improve future natural language representation in general

1.2 Goal

Goal:

- Improve on benchmark without task-specific engineering

Proposing:

- Intermediate representation extracted from large unlabeled dataset
  - using a single learning system.
- More general than any benchmarks
  - since no large body of linguistic knowledge is involved.
- Approach called “almost from scratch”

Section 2: Benchmark Tasks

- Part-Of-Speech Tagging (POS)
- Chunking
- Named Entity Recognition (NES)
- Semantic Role Labeling (SRL)

For each task, the state-of-the-art system that did **not** use external **labelled** data was chosen as the benchmark system in order to have a fair comparison based on the quality of a system and not the data used for the system.

As a task becomes more complex, more engineered features are needed to train on.

2.1: Part-Of-Speech Tagging

- Label each word in a text with its syntactic role
  - “He sat on the desk” would be labelled as singular noun, past tense verb, preposition, article
- Benchmark system: Toutanova et al. (2003)
  - 97.24% per-word accuracy
  - Used maximum entropy classifiers and inference in bidirectional dependency network
- Dataset: Wall Street Journal data

2.2: Chunking (Shallow Parsing)

- Label segments of a sentence with syntactic constituents (parts of a sentence that act as a single unit), then label words as in a chunk or begin a chunk
    - “The tall basketball player jumped over his smaller opponent” would have three chunks, two noun-phrases and a verb-phrase.
- Benchmark system: Sha and Pereira (2003)
    - 94.29% F1 Score
    - Used second-order random fields and a variety of features such as part-of-speech tags
- Dataset: CoNLL 2000

2.3: Named Entity Recognition

- Tag elements in a text with a category
    - “Kevin lives in Seattle” would be labelled as person, other, other, location.
- Benchmark system: Ando and Zhang (2005)
    - 89.31% F1 Score
    - Trained a model on NER and two auxiliary unsupervised tasks while permoign Viterbi decoding when testing
- Dataset: CoNLL 2003

2.4: Semantic Role Labeling

- Assigns a role to each syntactic constituent in a text
  - Typically assigns roles between ARG0 and 5 to each argument of a verb
  - “John ate the apple” would be tagged as ARG0, verb, ARG1.
- Benchmark system: Koomen et al. (2005)
  - 77.92% F1 Score
  - Used Winnow-like classifiers along with a decoding stage
- Dataset: CoNLL 2005

Section 3: The Networks

- Old approach: hand-design task-specific features, feed to SVM
  - Feature selection is task-dependent
  - Complex feature have high computational cost
- New approach: pre-process as little as possible, train NN in E2E fashion
  - NN learns feature extraction
  - Features automatically trained to be relevant to task

3.1: Notation

- Think of NN as a composition of functions

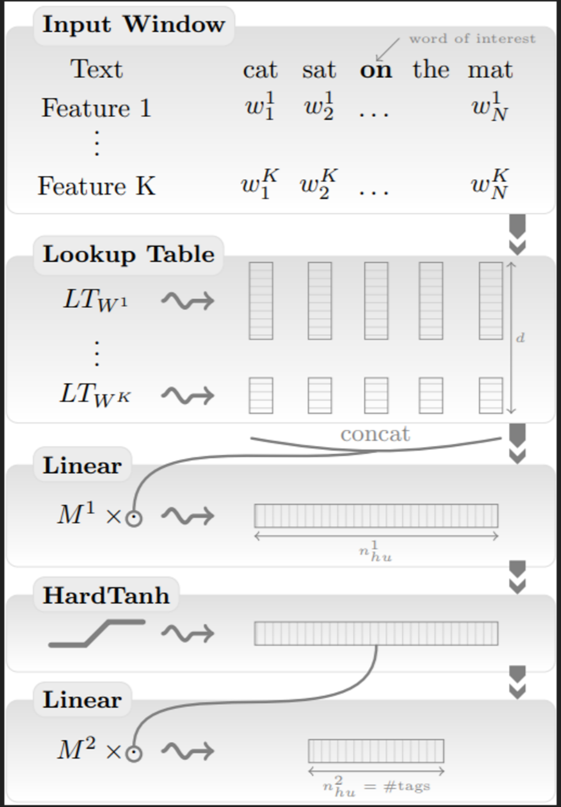
$$f_\theta(\cdot) = f_\theta^L(f_\theta^{L-1}(\ldots f_\theta^1(\cdot)\ldots))\,.$$

- Given matrix A, $[A]_{i,j}$ is value at row i, column j
- Window of size d^{win} around column i

$$\left[\langle A\rangle_i^{d_{win}}\right]^{\mathrm{T}} = \left([A]_{1,\,i-d_{win}/2} \cdots [A]_{d_1,\,i-d_{win}/2}\,, \ldots, [A]_{1,\,i+d_{win}/2} \cdots [A]_{d_1,\,i+d_{win}/2}\right)\,.$$

- Sequence of $\{x_1,x_2,\ldots,x_T\}$ written as $[x]_1^T$

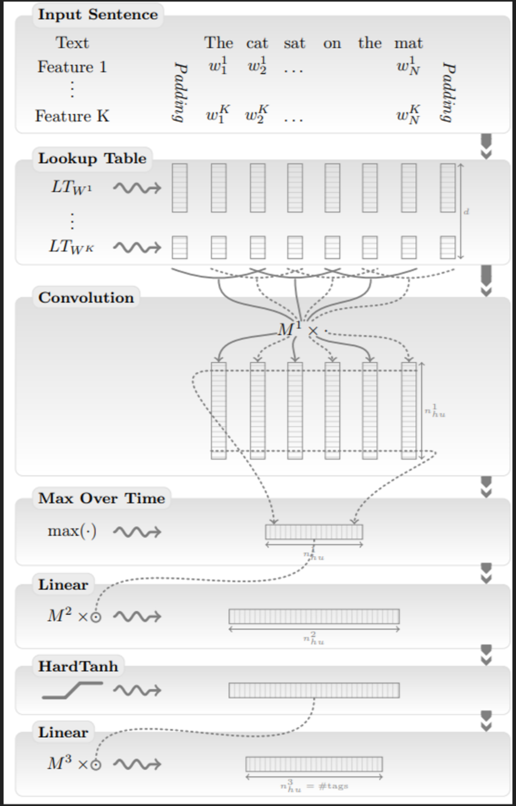
3.3.1: Window Approach



HardTanh Layer

$$\mathrm{HardTanh}(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } -1 <= x <= 1 \\ 1 & \text{if } x > 1 \end{cases}$$

3.3.2: Sentence approach



3.3.2: Sentence approach

- For SRL, add features for relative distance to verb considered and word being tagged

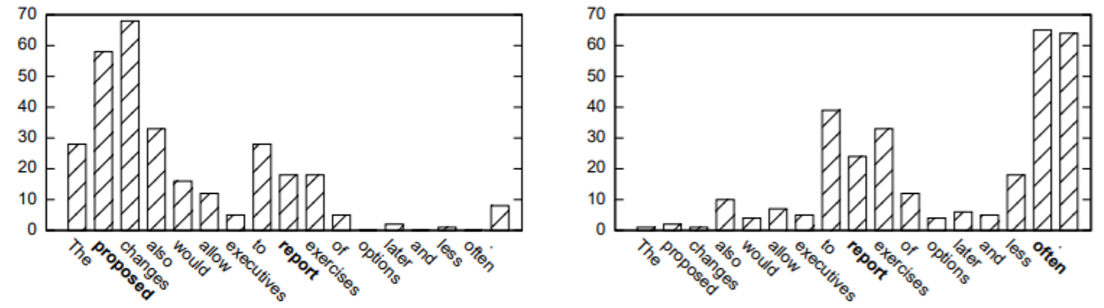
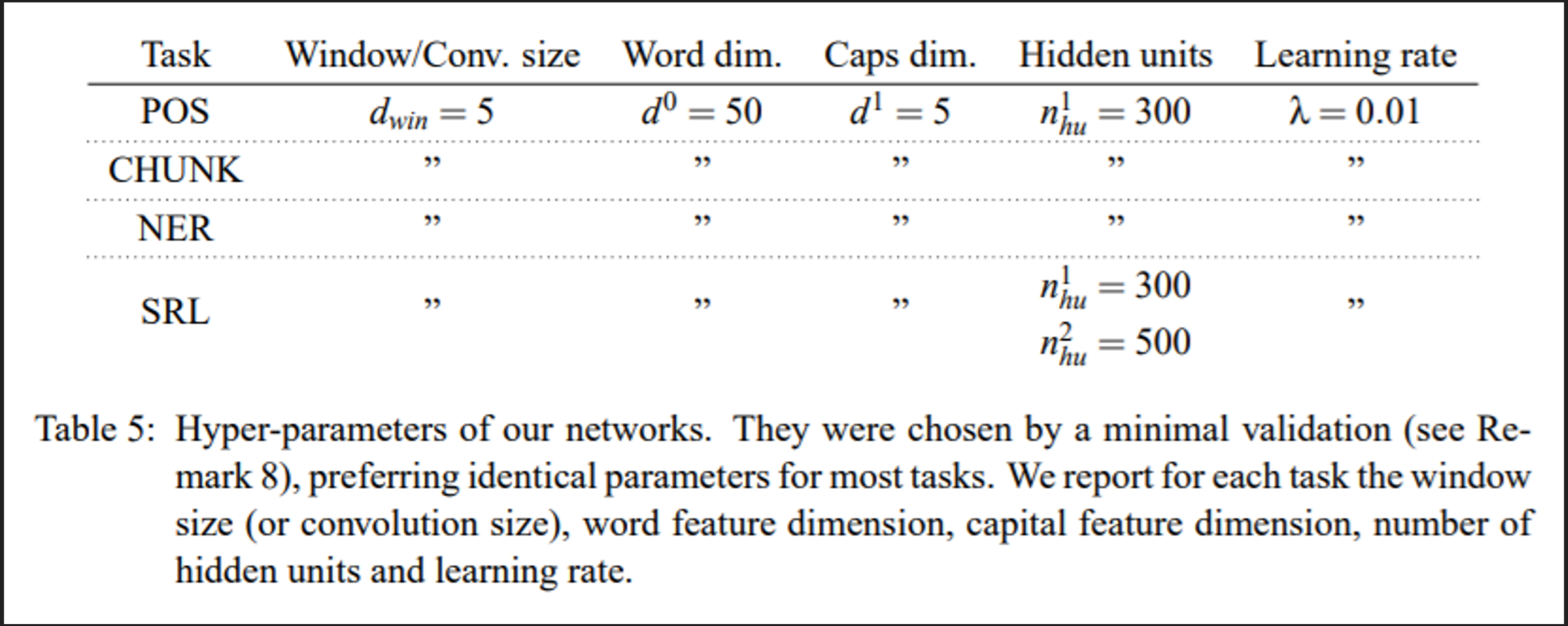


Figure 3: Number of features chosen at each word position by the Max layer. We consider a sentence approach network (Figure 2) trained for SRL. The number of "local" features output by the convolution layer is 300 *per word*. By applying a Max over the sentence, we obtain 300 features for the *whole sentence*. It is interesting to see that the network catches features mostly around the verb of interest (here "report") and word of interest ("proposed" (left) or "often" (right)).

3.4: Training

- Word-level Log-likelihood (WLL): Cross-entropy loss, we have covered that before
- Sentence-level Log-likelihood (SLL): Consider which “tag paths” are valid
    - Construct transition matrix representing validity of jumping from tag i to tag j in successive words. This is trained along with network.
    - Add transition scores, network scores when scoring tag paths.
- Trained using Stochastic Gradient Ascent

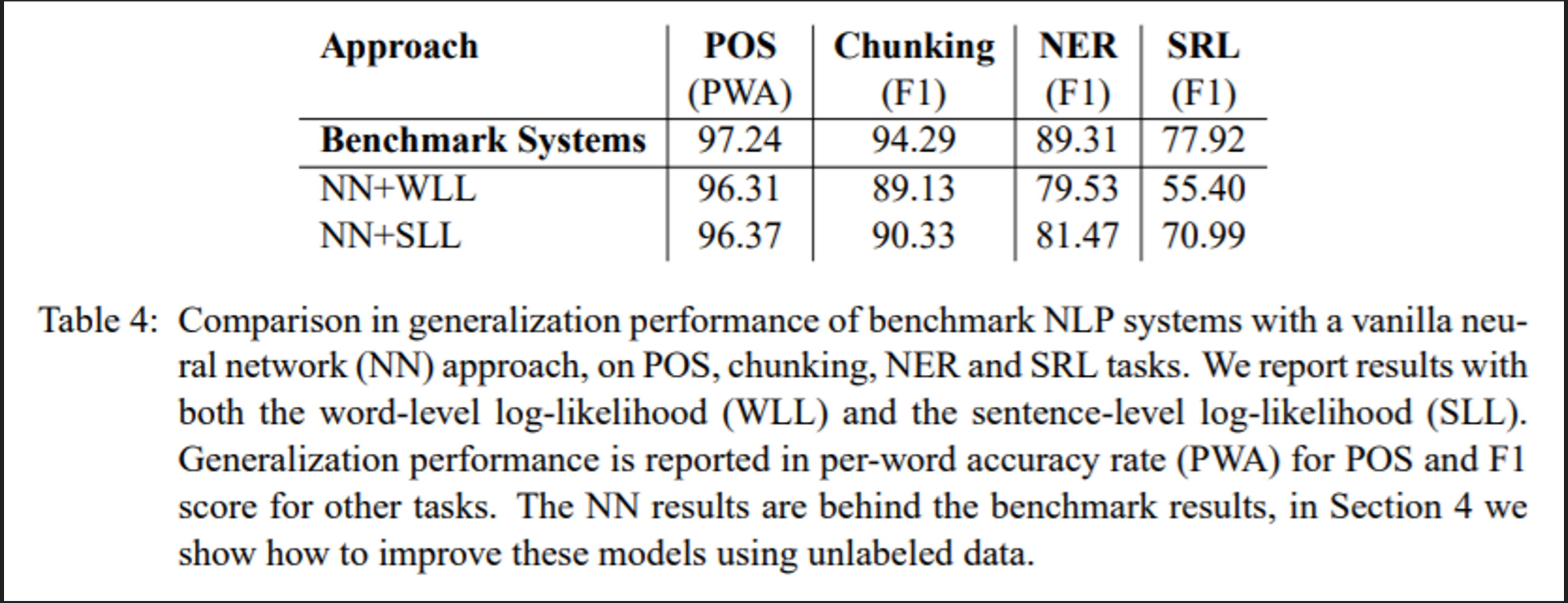
3.5: Supervised Training Results



Task	Window/Conv. size	Word dim.	Caps dim.	Hidden units	Learning rate
POS	$d_{win} = 5$	$d^0 = 50$	$d^1 = 5$	$n^1_{hu} = 300$	$\lambda = 0.01$
CHUNK	”	”	”	”	”
NER	”	”	”	”	”
SRL	”	”	”	$n^1_{hu} = 300$ $n^2_{hu} = 500$	”

Table 5: Hyper-parameters of our networks. They were chosen by a minimal validation (see Remark 8), preferring identical parameters for most tasks. We report for each task the window size (or convolution size), word feature dimension, capital feature dimension, number of hidden units and learning rate.

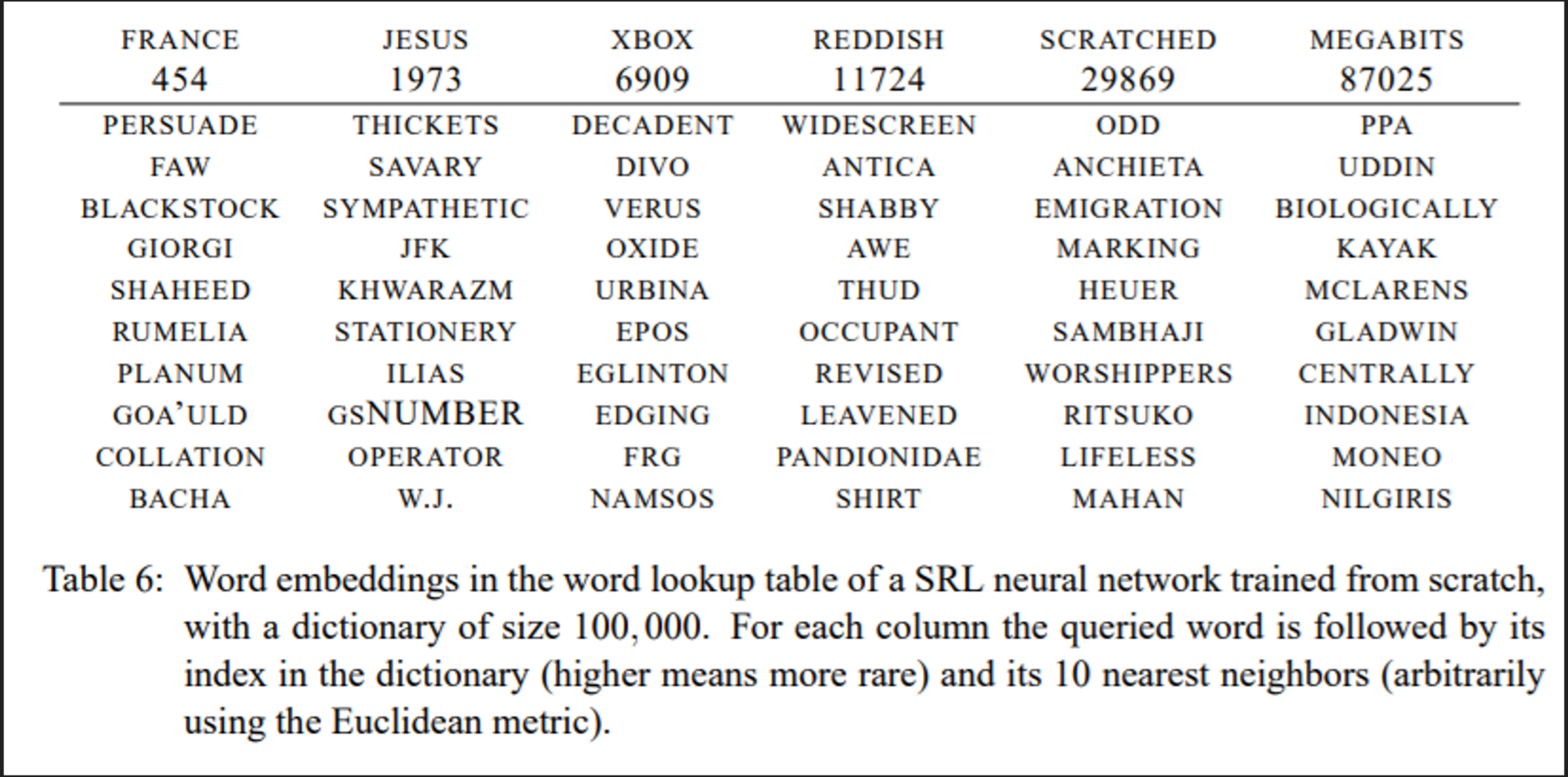
3.5: Supervised Training Results



Approach	**POS** (PWA)	**Chunking** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99

Table 4: Comparison in generalization performance of benchmark NLP systems with a vanilla neural network (NN) approach, on POS, chunking, NER and SRL tasks. We report results with both the word-level log-likelihood (WLL) and the sentence-level log-likelihood (SLL). Generalization performance is reported in per-word accuracy rate (PWA) for POS and F1 score for other tasks. The NN results are behind the benchmark results, in Section 4 we show how to improve these models using unlabeled data.

3.5: Supervised Training Results



FRANCE 454	JESUS 1973	XBOX 6909	REDDISH 11724	SCRATCHED 29869	MEGABITS 87025
PERSUADE	THICKETS	DECADENT	WIDESCREEN	ODD	PPA
FAW	SAVARY	DIVO	ANTICA	ANCHIETA	UDDIN
BLACKSTOCK	SYMPATHETIC	VERUS	SHABBY	EMIGRATION	BIOLOGICALLY
GIORGI	JFK	OXIDE	AWE	MARKING	KAYAK
SHAHEED	KHWARAZM	URBINA	THUD	HEUER	MCLARENS
RUMELIA	STATIONERY	EPOS	OCCUPANT	SAMBHAJI	GLADWIN
PLANUM	ILIAS	EGLINTON	REVISED	WORSHIPPERS	CENTRALLY
GOA'ULD	GSNUMBER	EDGING	LEAVENED	RITSUKO	INDONESIA
COLLATION	OPERATOR	FRG	PANDIONIDAE	LIFELESS	MONEO
BACHA	W.J.	NAMSOS	SHIRT	MAHAN	NILGIRIS

Table 6: Word embeddings in the word lookup table of a SRL neural network trained from scratch, with a dictionary of size 100,000. For each column the queried word is followed by its index in the dictionary (higher means more rare) and its 10 nearest neighbors (arbitrarily using the Euclidean metric).

Section 4: Lots of Unlabeled Data

- We want word embeddings with more information than the ones in the table of the SRL neural network in 3.5
- We will see how to improve our poor embeddings using large unlabeled data sets
- New embeddings to make new word lookup tables for our networks

4.1: Data Sets

- Entire English Wikipedia
  - Removed all paragraphs non-Roman characters and markups
  - Tokenized, 631 million words
  - Dictionary of 100,000 words from Wall Street Journal
- Reuters RCV1 data set
  - 221 million words
  - Extended dictionary to 130,000 words adding the 30,000 most common words in Reuters
  - We can determine if adding this data set can yield improvements

4.2: Ranking Criterion vs. Entropy Criterion

- Used data sets to train language models that output scores for text
  - Most trainable parameters are in the lookup tables
- Previous works: Bengio and Ducharme (2001), Schwenk and Gauvain (2002)
  - Estimate the probability of a word given previous words in the sentence
  - Suggests cross-entropy criterion
  - Computing normalization term is demanding given large dictionary
  - Neither work gave significant word embeddings
  - We can’t really use the 0.2 bit/character entropy difference between humans and n-gram models to learn grammar
  - Entropy criterion lacks dynamical range because its value is mostly determined by the most frequent phrases
  - To learn syntax, rare but legal phrases are no less significant than common phrases

4.2: Ranking Criterion vs. Entropy Criterion

- They propose a pairwise ranking approach
  - We do not want to emphasize a common phrase over a rare but legal phrase
  - Window network approach

$$\theta \mapsto \sum_{x \in \mathcal{X}} \sum_{w \in \mathcal{D}} \max\left\{0\,,\,1 - f_\theta(x) + f_\theta(x^{(w)})\right\}$$

- Okanohara and Tsujii (2007) used a similar approach but with binary classifications and a kernel classifier, not with word embeddings
- Smith and Eisner (2005), “negative” neighborhood

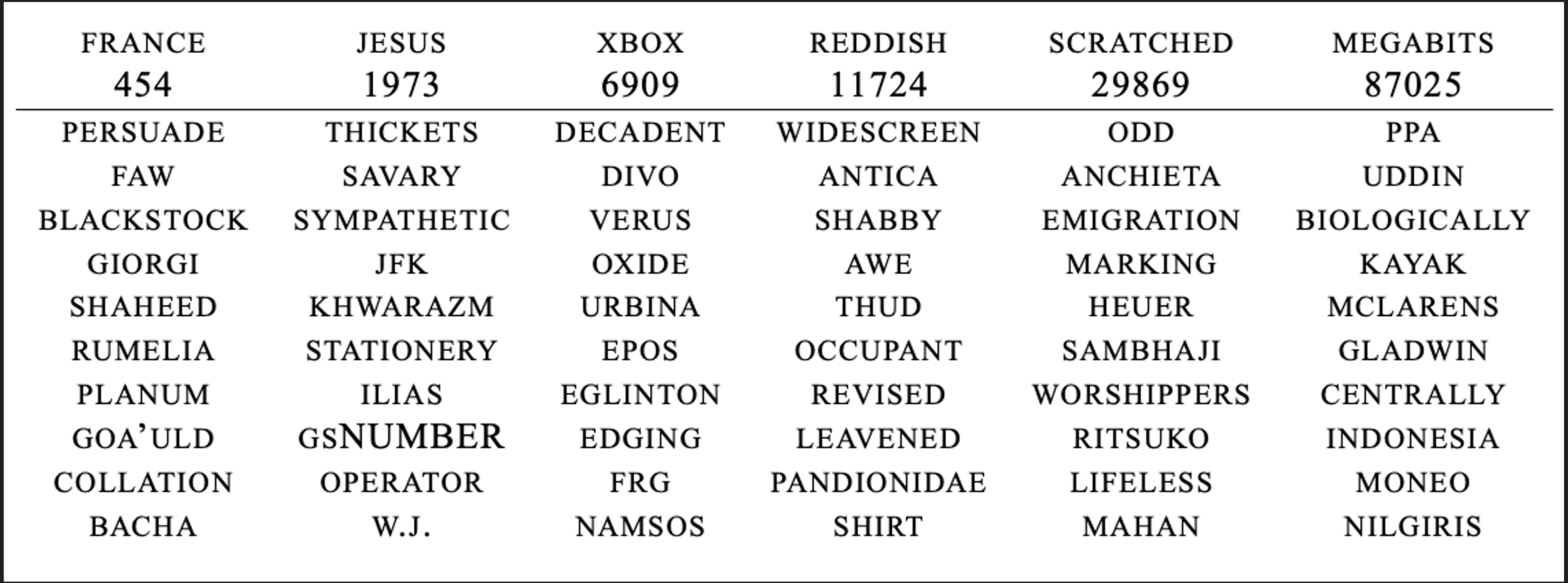
4.3: Training Language Models

- Trained on SGD of the ranking criterion, sampling sentence-word pairs each iteration
- Cannot tune global hyperparameters
- Initialize networks with embeddings from earlier networks
- Train a succession of networks using increasingly large dictionaries, each network being initialized with the embeddings of the previous network (sizes and switching times arbitrary)
- Breeding
  - Child networks initialized with parent’s embeddings and different training parameters based on past generation’s success (learning rate, word embedding dimensions, # of hidden dimensions, etc.)

4.3: Training Language Models

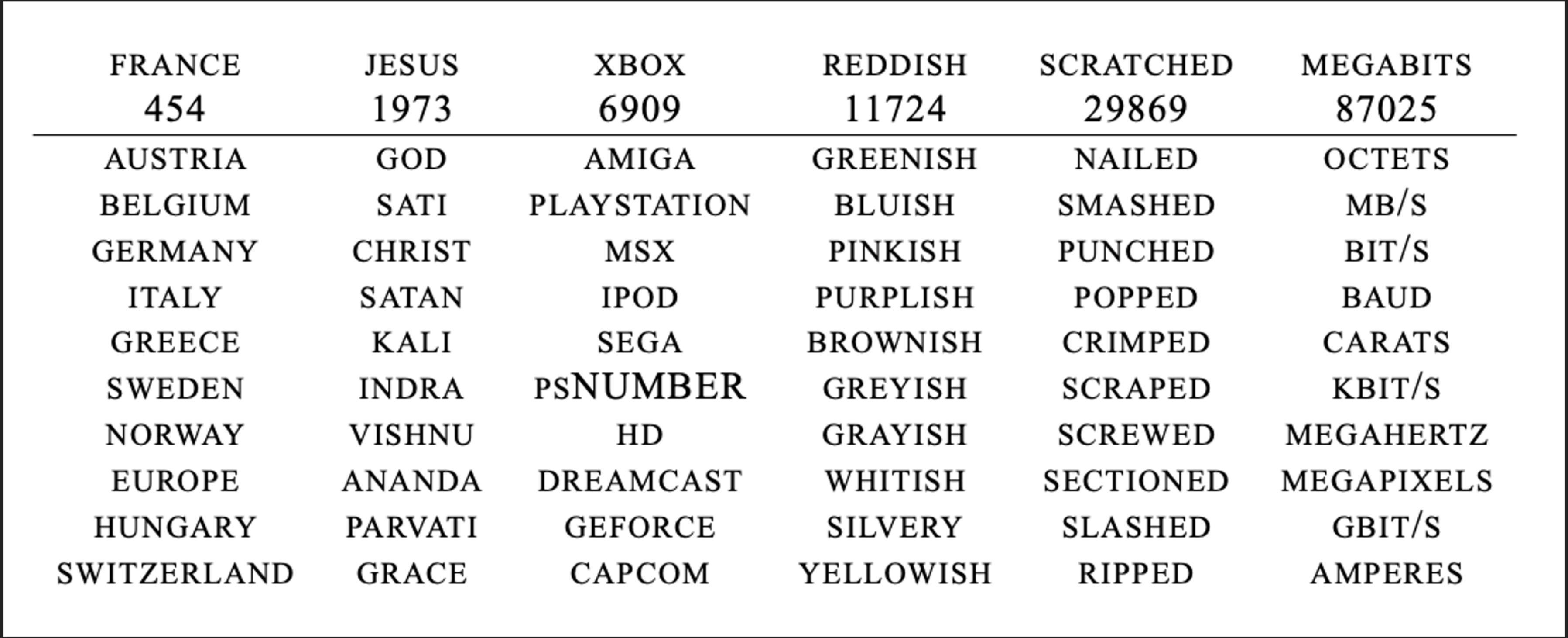
- LM1
  - Window size 11, hidden layer with 100 units
  - Trained on Wikipedia with dictionary sizes of 5k, 10k, 30k, 50k, and 100k most common Wall Street Journal words
  - 4 weeks
- LM2
  - Initialized with LM1 embeddings, same dimensions as LM1
  - Trained for another 3 weeks on Wikipedia+Reuters with 130k-word dictionary

4.4: Embeddings (Before)



FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
454	1973	6909	11724	29869	87025
PERSUADE	THICKETS	DECADENT	WIDESCREEN	ODD	PPA
FAW	SAVARY	DIVO	ANTICA	ANCHIETA	UDDIN
BLACKSTOCK	SYMPATHETIC	VERUS	SHABBY	EMIGRATION	BIOLOGICALLY
GIORGI	JFK	OXIDE	AWE	MARKING	KAYAK
SHAHEED	KHWARAZM	URBINA	THUD	HEUER	MCLARENS
RUMELIA	STATIONERY	EPOS	OCCUPANT	SAMBHAJI	GLADWIN
PLANUM	ILIAS	EGLINTON	REVISED	WORSHIPPERS	CENTRALLY
GOA'ULD	GSNUMBER	EDGING	LEAVENED	RITSUKO	INDONESIA
COLLATION	OPERATOR	FRG	PANDIONIDAE	LIFELESS	MONEO
BACHA	W.J.	NAMSOS	SHIRT	MAHAN	NILGIRIS

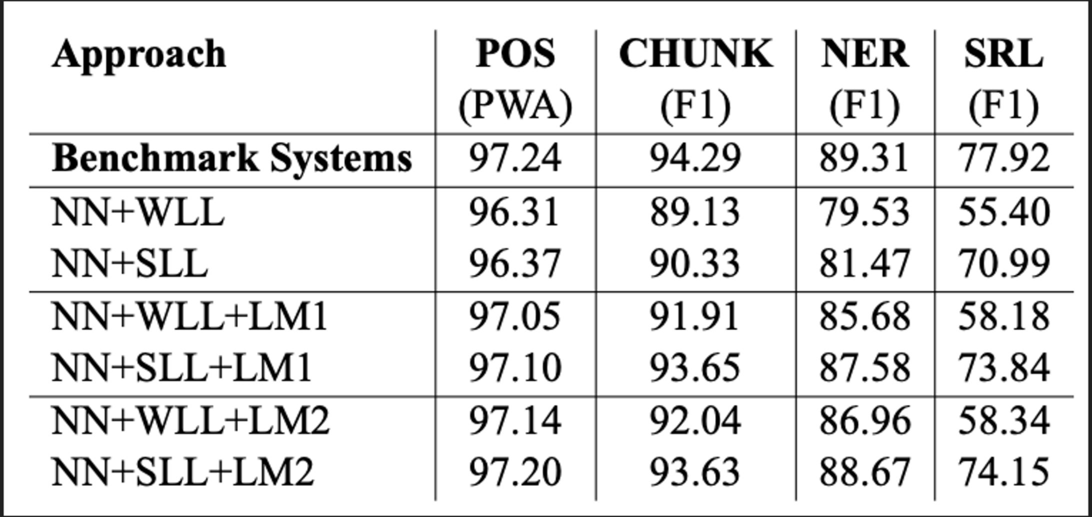
4.4: Embeddings (After - LM1)



FRANCE 454	JESUS 1973	XBOX 6909	REDDISH 11724	SCRATCHED 29869	MEGABITS 87025
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

4.5: Semi-supervised Benchmark Results

- Semi-supervised learning
  - Ad hoc
  - Self-training (pseudo-labels)
  - Parameter-sharing



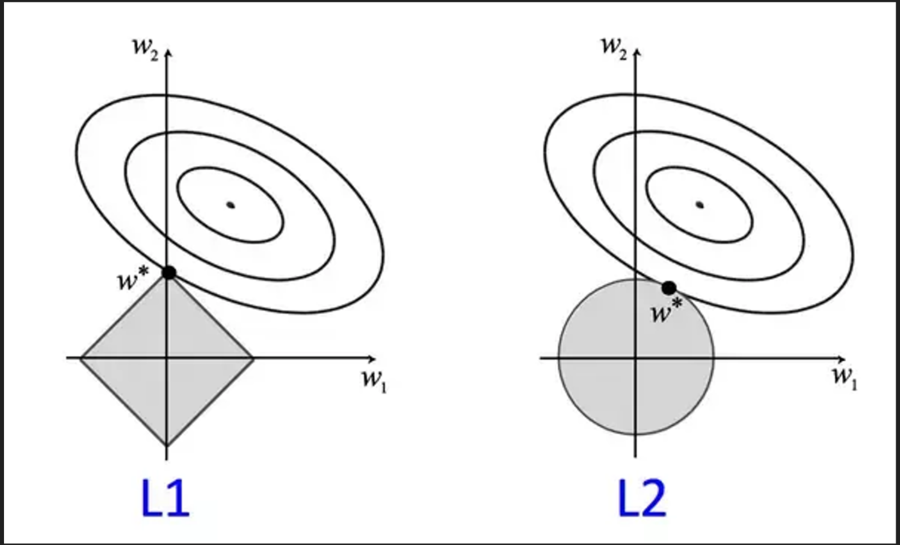
Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99
NN+WLL+LM1	97.05	91.91	85.68	58.18
NN+SLL+LM1	97.10	93.65	87.58	73.84
NN+WLL+LM2	97.14	92.04	86.96	58.34
NN+SLL+LM2	97.20	93.63	88.67	74.15

4.6: Ranking and Language

- Syntax is a prerequisite for semantic role labeling
  - Existing semantic role labeling systems use parse trees, parsers know prior info about syntax
  - They can’t use parse trees with unlabeled data
- Difficult to see how ranking criterion can obtain this information
- Ranking similar to operator grammars (Harris, 1968), which defines a ranking criterion when testing if two sentences are semantically related by a transformation
- They conclude that ranking criterion has the potential to extract strong syntactic and semantic information
- Our language models are too restrictive for our goals

5. Multi-Task Learning

- Features trained for one task can be useful for related tasks
  - Already seen this with unsupervised word embeddings
  - Has since become very popular in Computer Vision and RL
- Can be seen as a form of regularization



5.1 Joint Decoding versus Joint Training

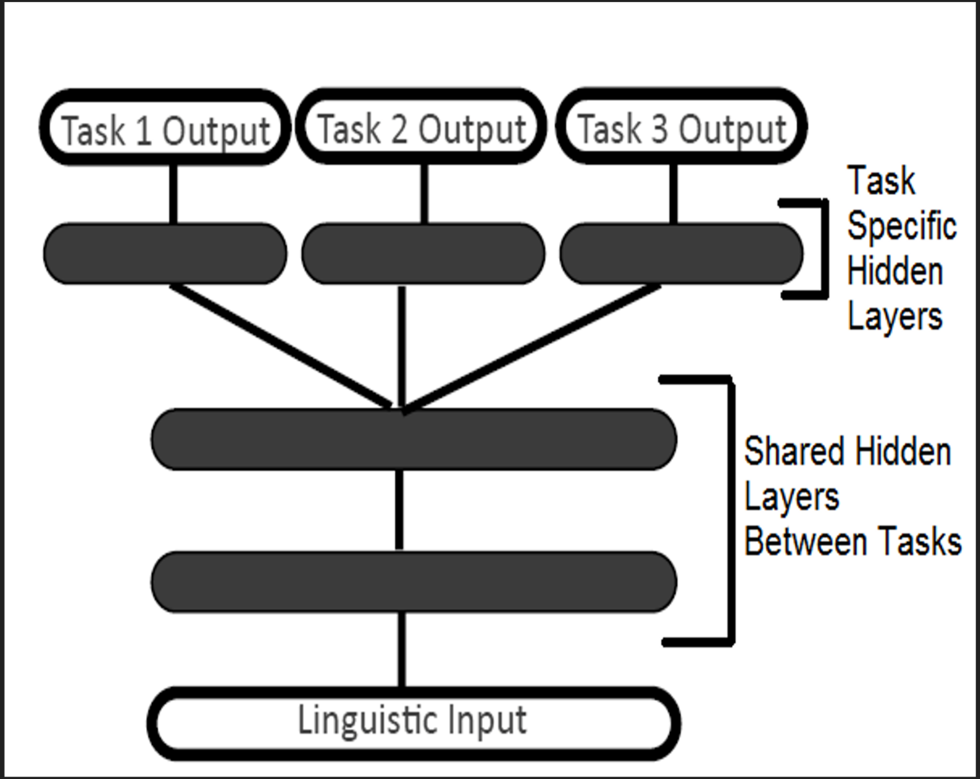
Joint Decoding

- Probabilistic framework for inference across models
- Used in multi-modal domains like speech recognition

5.1 Joint Decoding versus Joint Training

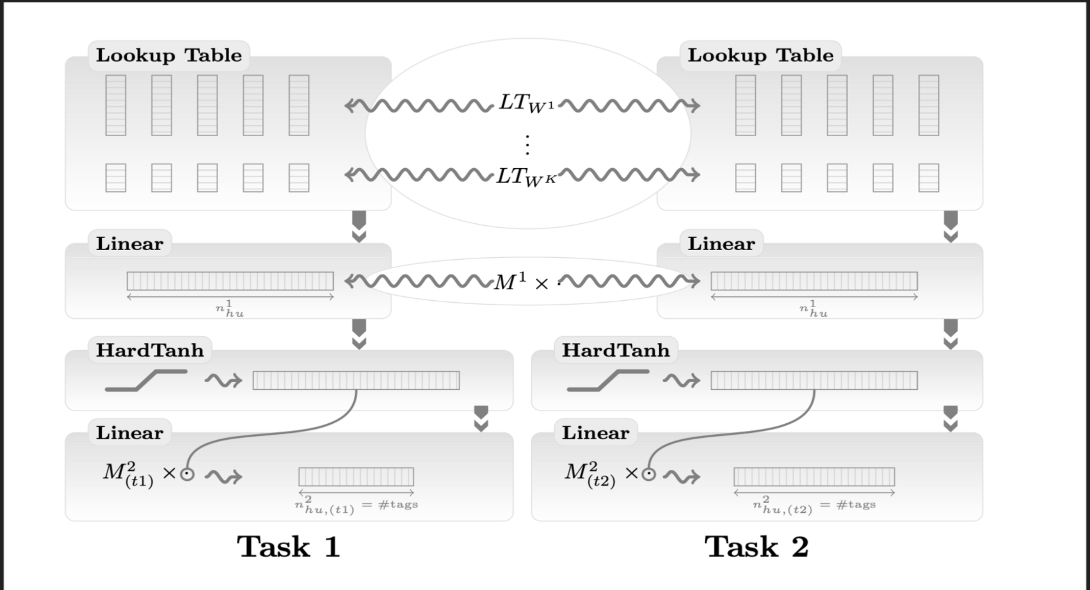
Joint Training

- One network with multiple outputs
  - Need labels for each task for every input
  - Loss as linear combination of tasks
- What to do when labels correspond to different training sets?
  - Iterative gradient updates
  - Theoretically unstable, but works in practice

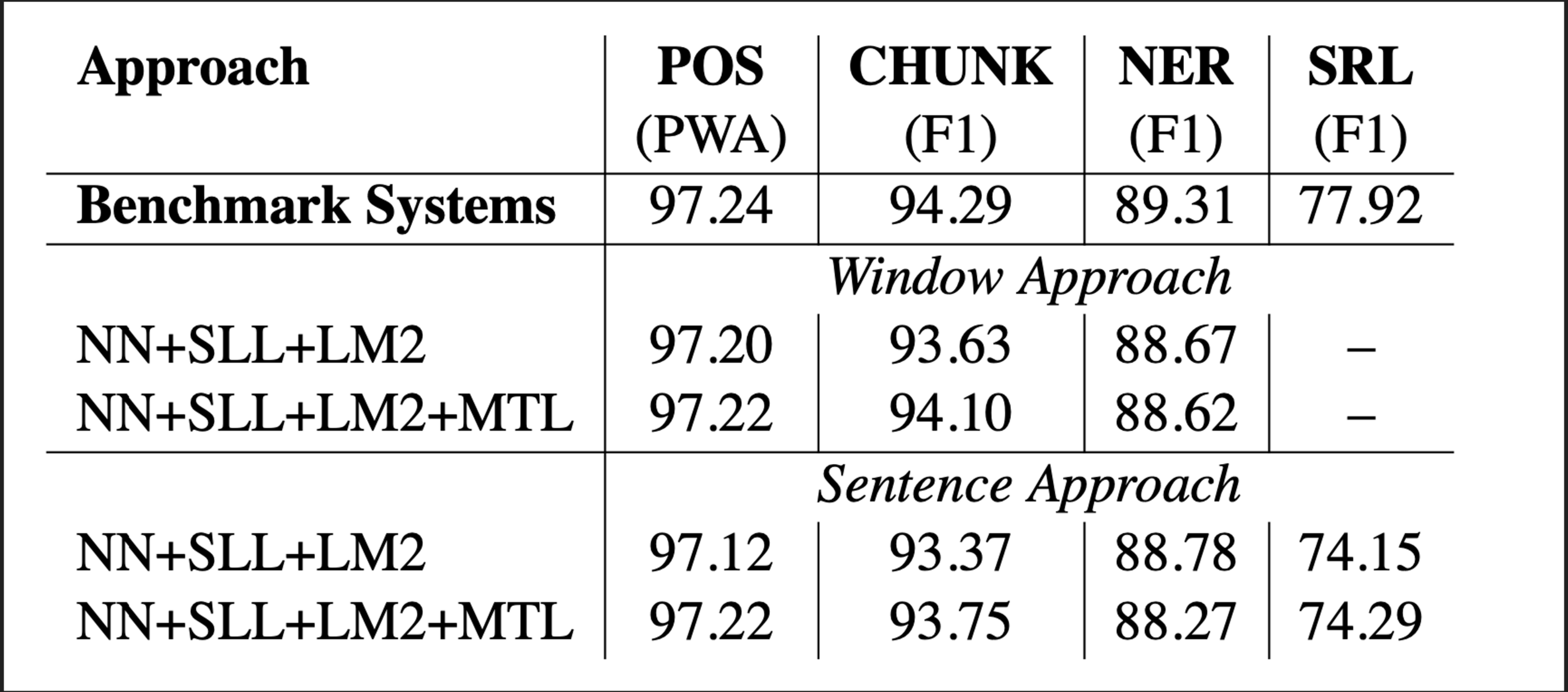

Task 1 Output
Task 2 Output
Task 3 Output
Task Specific Hidden Layers
Shared Hidden Layers Between Tasks
Linguistic Input

5.2 Multi-Task Benchmark Results

- In this work specifically:
  - Shared lookup tables
  - Shared first linear layer in window architecture, first conv layer in sentence architecture
  - Alternate between tasks for gradient updates



5.2 Multi-Task Benchmark Results



Approach	POS (PWA)	CHUNK (F1)	NER (F1)	SRL (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
	Window Approach			
NN+SLL+LM2	97.20	93.63	88.67	–
NN+SLL+LM2+MTL	97.22	94.10	88.62	–
	Sentence Approach			
NN+SLL+LM2	97.12	93.37	88.78	74.15
NN+SLL+LM2+MTL	97.22	93.75	88.27	74.29

Section 6: ***The Temptation***

- Results so far have been (almost*) from scratch, as originally intended
  - **_almost_*: since we preprocessed the data into raw words-- if we really worked from scratch, we would start from characters (or pictures of characters!)
- This has disregarded a large amount of linguistic knowledge
- We’ve shown that, by using large unlabeled datasets, we can still get near state-of-the-art performance
- So how much better can we get if we increase “task-specific engineering” using known techniques from NLP literature?

6.1: Suffixes

- Statistically, **suffixes** are strong predictors of word syntactic function
- This is useful for part-of-speech taggers (the “POS system”)
- Let’s add discrete word features that represent the last two characters of each word
  - Size of suffix dictionary: **455**
- Improves POS tagging score from **97.20** to **97.29 + (<u>0.09</u>)**

6.2 Gazetteers

- A **“gazetteer”** is a dictionary containing well-known named entities
- As you might imagine, this is useful for the **named entity recognition (NER) task**
- Four gazetteer categories: (1) locations, (2) person names, (3) organizations, (4) miscellaneous
- Add 4 additional feature for each word that are “on” and “off” if the word is found in each of four categories
- Increases NER score from **88.67** to **89.59 (+0.92)**

6.3 Cascading

- **Cascading**: using features outputted by one task as input for another task
- *"Conventional NLP systems often use features obtained from the output of other preexisting NLP systems"*

- Can we improve CHUNK and NER tasks by adding POS tags as features?
- Yes! Improvement on CHUNK from 93.63 to 94.32 (**+0.69**)

6.3 Ensembles

- **Ensemble models** combine the predictions of multiple diverse models to predict an outcome
- Constructing ensembles of classifiers is the best way to trade computational efficiency for generalization performance
- Many NLP systems achieve state-of-the-art performance by combining the outputs of multiple classifiers
- Because neural networks are nonconvex, training runs with different initializations usually give different solutions

6.3 Ensembles [cont’d]

- Try ten training runs on each POS, CHUNK, NER task, with ten different initializations
- Voting leads to a small improvement (.1 to .3) in average network performance
- They also tried adding a linear layer to the outputs of the classifiers and training that way: this did not outperform the simple voting
- Performance variability among the networks is not very large, anyway

6.5 Parsing

- Some past researchers have argued that syntactic parsing is *necessary* for the SRL (semantic role labeling) task

- So far, we've gotten close to state-of-the-art on SRL without parse trees at all 😲

- Add parse tree features to the system:

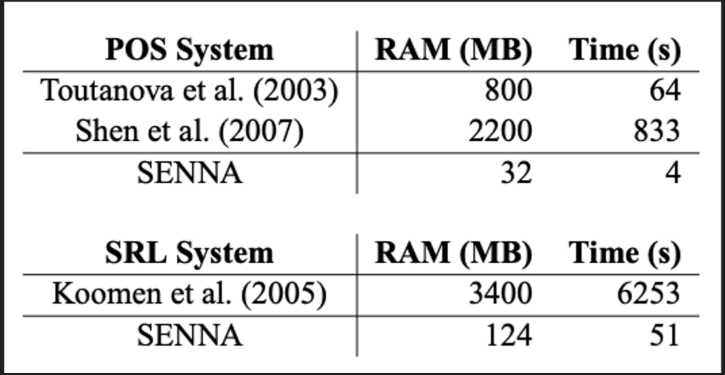
  Increase score from **74.15** to **77.92** (+**3.8%**)

6.6 Word Features

- They created their word embeddings via language modelling, instead of using previously established algorithms (like “Brown clustering”)
- A natural question: how to these embeddings compare? Is one more useful than the other?
- Answer: the new word embeddings do better! (very slightly)

6.7 Engineering a Sweet Spot

- Implement a standalone version of the architecture and just try to get the best scores possible
- They engineered their system from scratch and it's over 200x faster than the SOTA system and uses much less RAM



POS System	RAM (MB)	Time (s)
Toutanova et al. (2003)	800	64
Shen et al. (2007)	2200	833
SENNA	32	4

SRL System	RAM (MB)	Time (s)
Koomen et al. (2005)	3400	6253
SENNA	124	51

wow

Section 7: Critical Discussion

- Why abandon all the pre-existing knowledge of engineering NLP features and instead pursue end-to-end learning?
  - Task-specific features do not transfer well to other tasks → Cannot have a method that generalize well to all tasks.
  - Want something generalizable
- Why neural networks?
  - It’s able to learn hidden representation of words.
  - Learning algorithm scales linearly, which allows it to take advantage of hardware advancements.

Section 8: Conclusion

Contribution:

- Multilayer neural network that can learn to handle a number of NLP tasks with both speed and accuracy.
- More broadly, an end-to-end learning system that can learn useful features from unlabeled data set instead of relying on engineered features.