Challenges in (machine) learning from textual software artifacts

Andrian (Andi) Marcus
seers group
UT DALLAS | Erik Jonsson School of Engineering and Computer Science

Software Engineering
for
Machine Learning Applications

Machine Learning Applications for Software Engineering

Source: utdallas.edu


SoftwarE Evolution ReSearch group
KING SAUNA
King Spa
Dallas, TX
Oscar Chaparro
Juan Manuel Florez

SEERS alumni

Denys Poshyvanyk
William and Mary

Sonia Haiduc
Florida State U.

Laura Moreno
Colorado State U.

Jairo Aponte
U. Nacional de Colombia

Research interests and goals

Information about the software and how it relates to code

- domain information, design rationale, etc.
- present in textual software artifacts

Help developer (better) develop (better) software

- we are not building “intelligent” systems
- AI/ML is just part of the solutions
- we are building automated assistants for (intelligent) developers
- we can tolerate some failure and some lack of trust

Research work

Happy users of ...

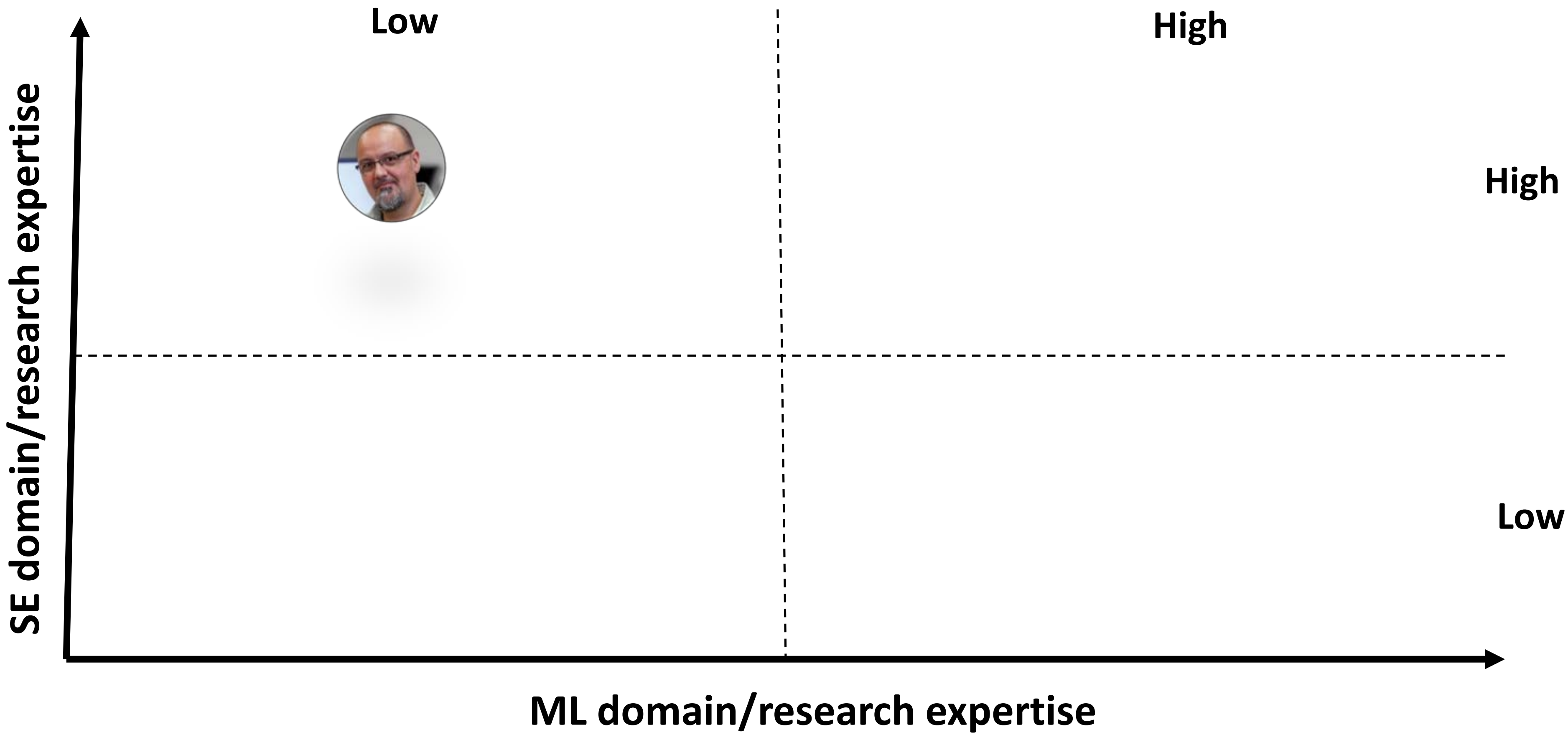
Information retrieval, for:

Concept/feature/bug localization	Bug triage
Traceability link recovery	Defect prediction
Software documentation generation	Impact analysis
Code quality and refactoring	Reverse engineering

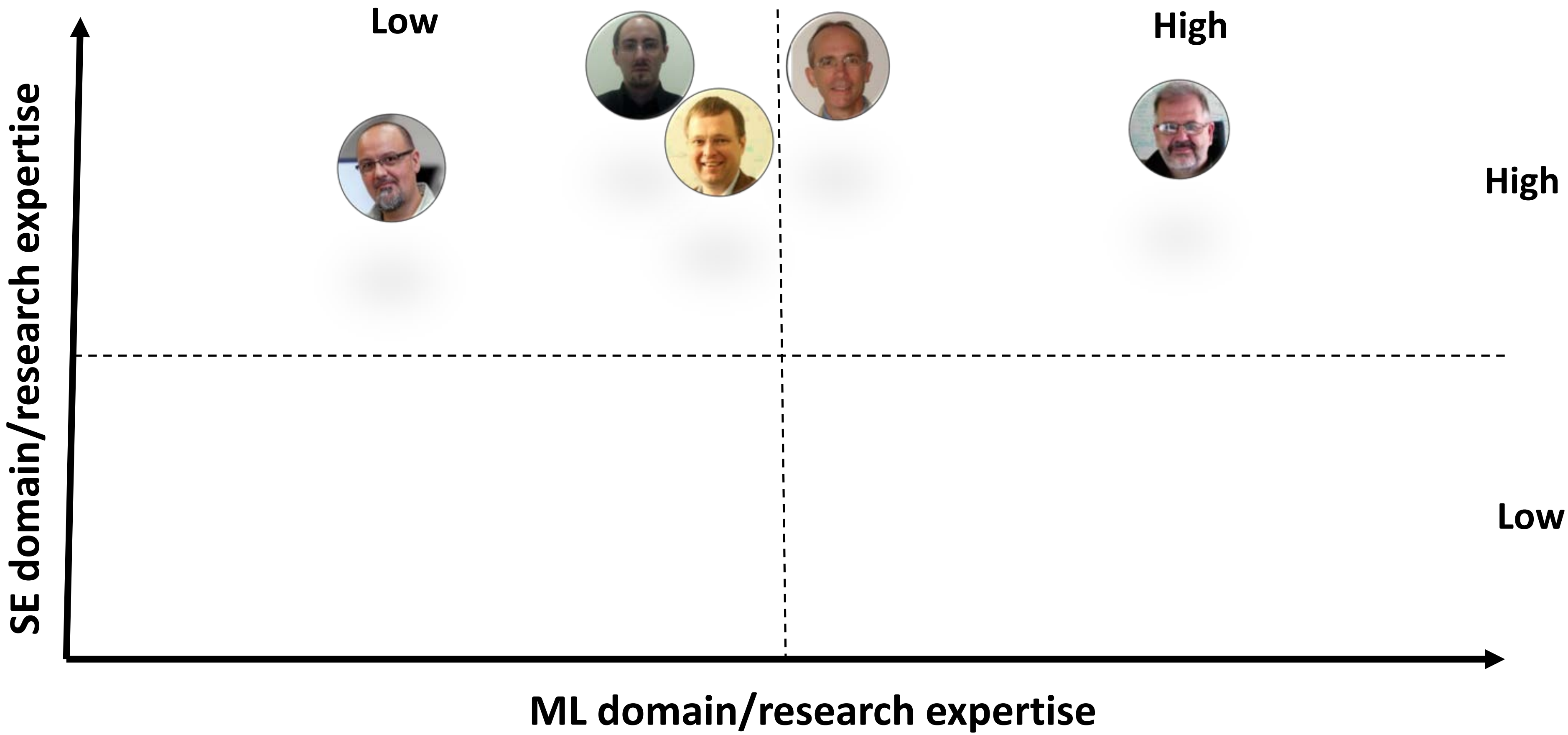
Machine learning , for:

Query improvement for code retrieval	Bug triage
Reverse engineering	Defect prediction

Where am I?


Low
High
High
Low
SE domain/research expertise
ML domain/research expertise

My SEMLA'18 collaborators



Our experience with ML in SE

Reverse engineering legacy code, code summarization

- clustering, heuristics

Defect prediction

- logistic regression, Bayes, classification trees, transfer learning

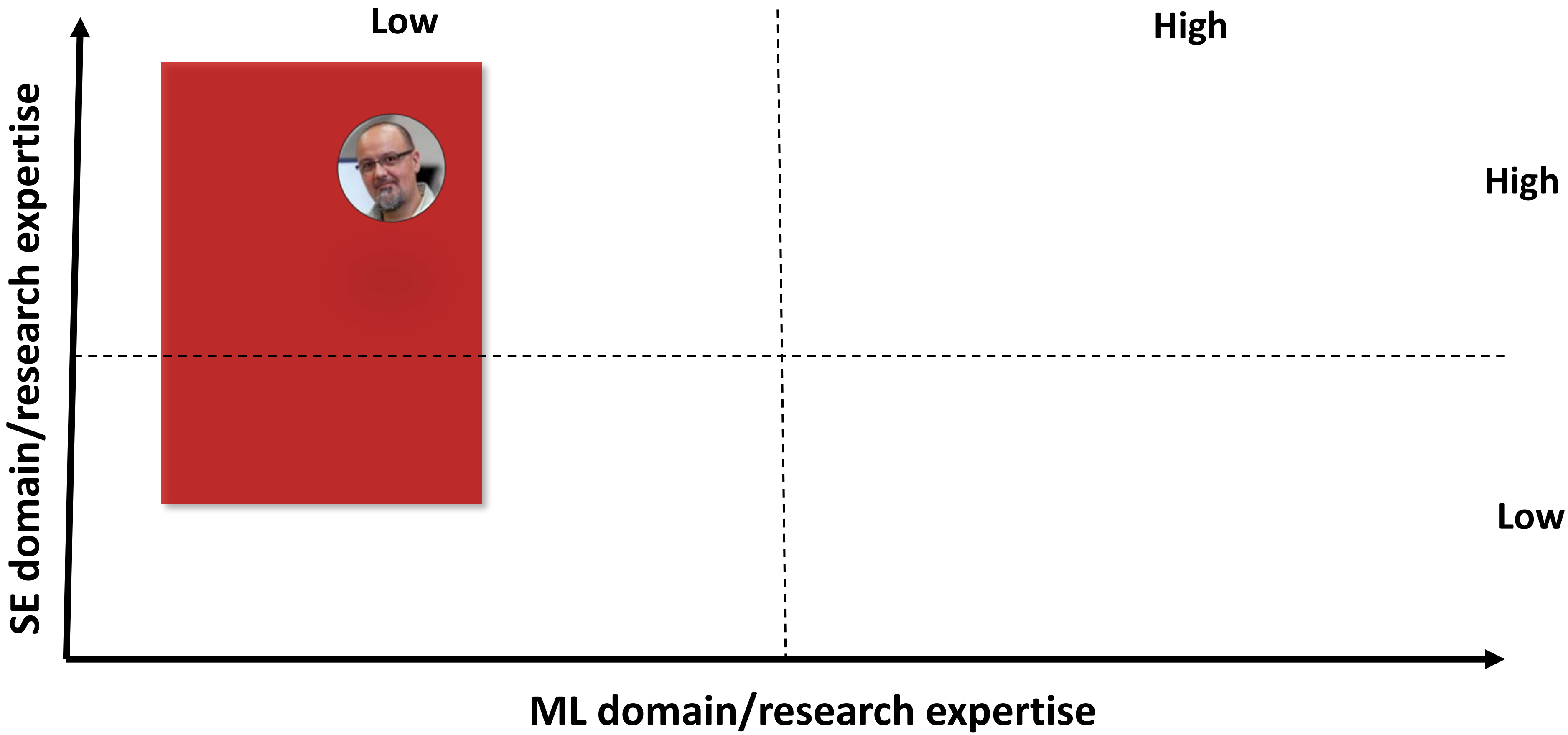
Query quality and reformulation for software retrieval

- classification trees

Identification of information in bug reports

- support vector machine, heuristics

Challenges as an (average) ML user in SE



How did we choose an ML technique?

Our expert collaborator said so

- reuse experience and expertise
- educational experience for us
- configuration and rationale for free
- requires a leap of faith

Collaborators

Tim Menzies, Max Di Penta, Denys Poshyvanyk, Sonia Haiduc, Laura Moreno, Giulio Antoniol, Gabriele Bavota, Gerardo Canfora, Giuseppe Scanniello, Rudolf Ferenc, Tibor Gyimothy, Vincent Ng, etc.

How did we choose an ML technique?

The same as previous work against which we compared

- the focus of the research is on the features
- reuse the experience of previous work
  – not always easy, poorly documented
- not always the best

How did we choose an ML technique?

Tried several and kept the one that has best results

- hard to decide which one IS the best
  - not always the same winner across data sets
- hard to explain why the best is best – usually guess
- the combination of techniques X parameters is huge
  - tough choices to make
- Relates to David Parnas' "lazy way"

How did we choose an ML technique?

Decision trees – the easy choice

- low configuration headache
- reasonable guidelines in training data selection
- relatively easy to explain the results
  - – which features matter most
- not always the best

Learning from bug reports

End user bug reports contain descriptions of:

observed behavior (OB), expected behavior (EB), steps to reproduce (S2R)

EB (65%) and S2R (49%) are often missing

Automatically detect the absence of EB and S2R

Chaparro, O., Lu, J., Zampetti, F., Moreno, L., Di Penta, M., Marcus, A., Ng, V.,
"Detecting missing information in bug descriptions",
***Joint Meeting on the Foundations of Software Engineering (ESEC/FSE 2017)*, pp. 376-387.**

Discourse patterns in bug descriptions

Tagged 2,900 bug reports

EB is described using 31 discourse patterns

S2R is described using 33 discourse patterns

Pattern code: S_EB_SHOULD
Description: sentence using the modals "should" or "shall" with no preceding predicates that use negative auxiliary verbs
Rule: [subject] should/shall (not) [complement]
Example: [*Apache*] ***should*** [*make an attempt to print the date in the language requested by the client*] (from Httpd 40431)

Machine learning

We used SVM

at the NLP expert (Vincent Ng) recommendation

Part of the labeled data was used for parameter calibration

The rest for intrinsic evaluation

Detecting missing EB

Approach	Strategy or Features	EB		
		Avg. Prec.	Avg. Recall	Avg. F_1
DeMIBuD-R	-	86.0%	85.9%	85.9%
DeMIBuD-H	all patterns	96.7%	46.1%	62.2%
DeMIBuD-H	no ambiguous patterns	95.1%	76.6%	84.7%
DeMIBuD-ML	pos	73.8%	93.1%	82.0%
DeMIBuD-ML	*n*-gram	75.1%	97.6%	84.7%
DeMIBuD-ML	pos + *n*-gram	76.0%	95.1%	84.2%
DeMIBuD-ML	patterns	85.9%	93.2%	89.4%
DeMIBuD-ML	patterns + pos	77.9%	92.9%	84.6%
DeMIBuD-ML	patterns + *n*-gram	76.9%	97.0%	85.6%
DeMIBuD-ML	pos + patterns + *n*-gram	76.8%	95.8%	85.1%

Detecting missing S2R

Approach	Strategy or Features	S2R		
		Avg. Prec.	Avg. Recall	Avg. F_1
DeMIBuD-R	-	63.3%	$\boxed{92.4\%}$	74.3%
DeMIBuD-H	all patterns	84.5%	31.0%	44.3%
DeMIBuD-H	no ambiguous patterns	$\boxed{81.6\%}$	38.5%	51.2%
DeMIBuD-ML	pos	60.8%	75.8%	66.8%
DeMIBuD-ML	*n*-gram	66.4%	$\boxed{83.4\%}$	73.4%
DeMIBuD-ML	pos + *n*-gram	65.3%	79.2%	71.1%
DeMIBuD-ML	patterns	63.5%	80.3%	70.7%
DeMIBuD-ML	patterns + pos	65.4%	76.0%	69.9%
DeMIBuD-ML	patterns + *n*-gram	69.2%	83.0%	$\boxed{74.9\%}$
DeMIBuD-ML	pos + patterns + *n*-gram	67.2%	80.9%	73.0%

Training data

Need expertise for labelling data (i.e., bug reports)

- cannot use Amazon Mechanical Turk or crowdsourcing
- very high cost per data point

Evaluation and application

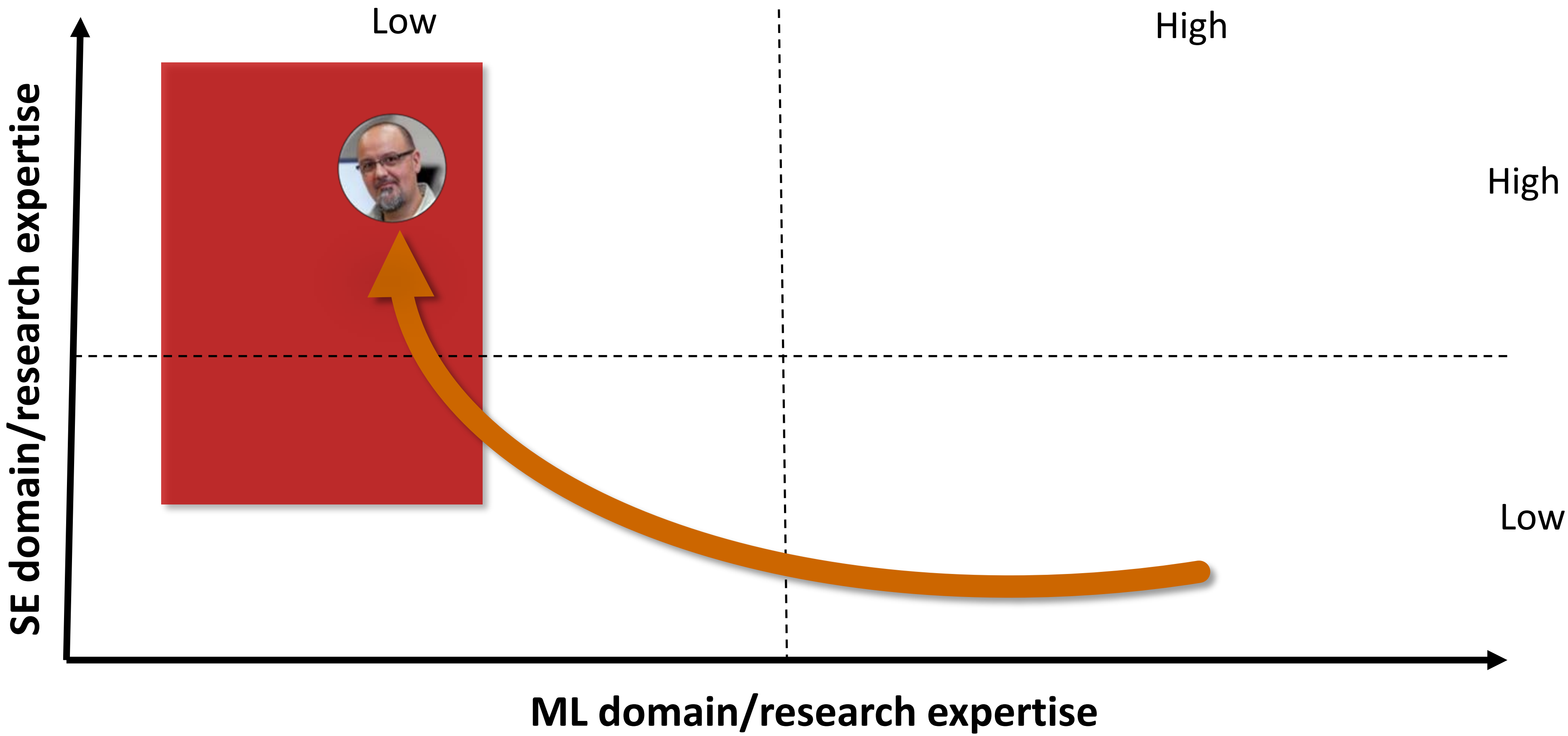
Extrinsic evaluation too costly

- needs integration with additional techniques
- the classification is often just an intermediary step of a solution

Cost of producing the training data limits applicability, despite better results than the heuristic based approach

Cannot infer explanations based on the NL features (i.e., pos)

What do I want from the ML experts?



Guidelines

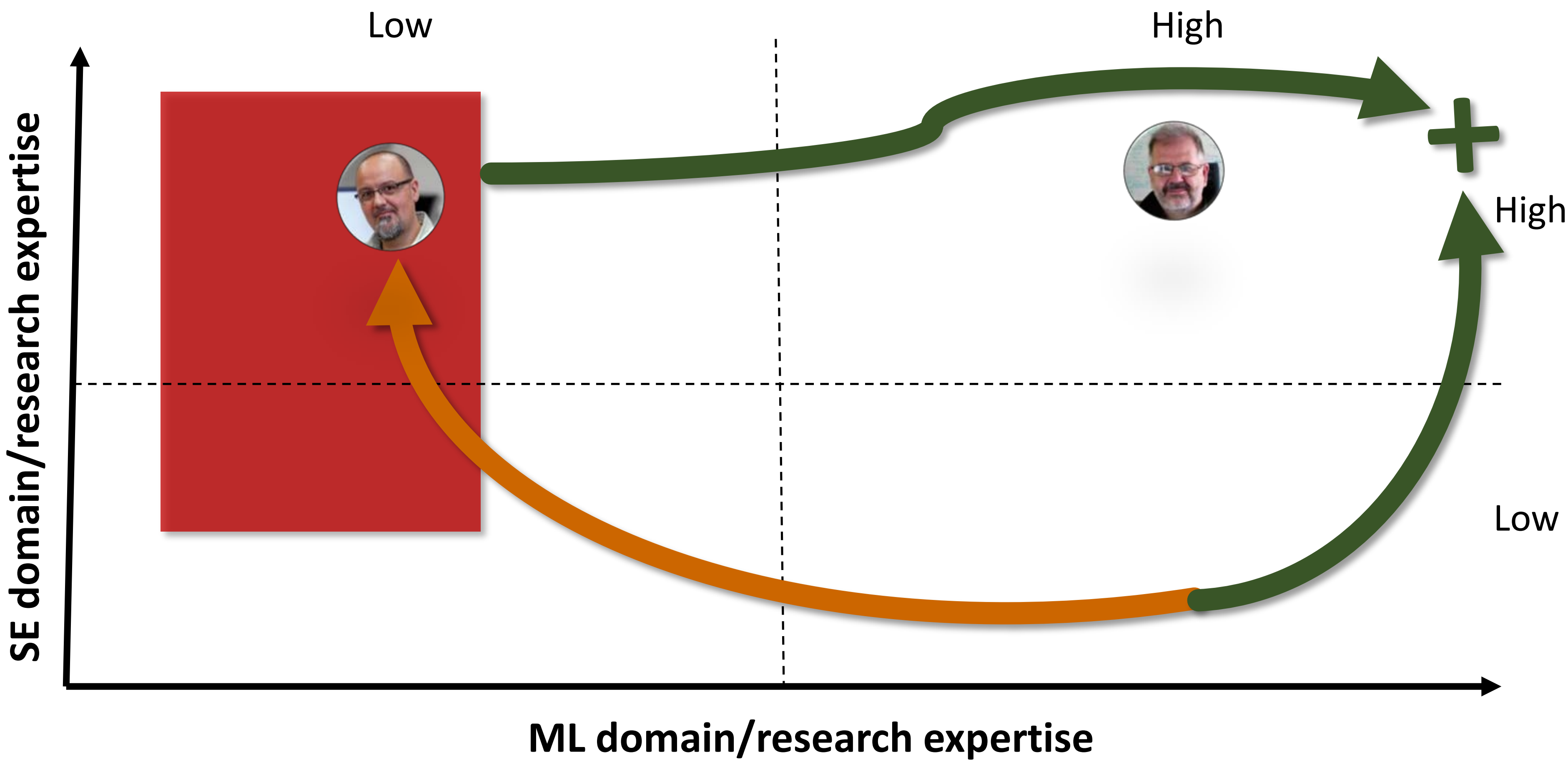
Which ML model is best for which type of data?

What are the optimal parameters?

How much training data?

What distribution should the training data have?

How can we beat Tim Menzies?



Working across computing disciplines: ML + SE

Very hard in academia

- Publish or perish
  - incremental results are favored
- Student training
  - expertise in two areas take much longer than Ph.D. time
- Scholarship is recognized differently across research areas
  - where should we published
- Contributions are different
  - adding to SE, but not ML
- Cost of long-term collaborations
  - easier to go on your own, after a while

In the end, we wish for …

ML models that perform well, are cheap to train, and easy to explain.

Guidelines from ML experts to help us with training data, configurations, and model selection.

Easier, long-term collaborations between SE and ML researchers/experts.

DysDoc3 - https://dysdoc.github.io/

In the end, we wish for …

ML models that perform well, are cheap to train, and easy to explain.

Guidelines from ML experts to help us with training data, configurations, and model selection.

Easier, long-term collaborations between SE and ML researchers/experts.