Adversarial Generation of Real-time Feedback with Neural Networks for Simulation-based Training

Xingjun Ma, Sudanthi Wijewickrema, Shuo Zhou, Yun Zhou, Zakaria Mhammedi, Stephen O'Leary, James Bailey

xingjunm@student.unimelb.edu.au

School of Computing and Information Systems
The University of Melbourne

Presenter: Xingjun Ma
24 August, 2017

Table of contents

1. Background
2. Problem Definition
3. Proposed Method
4. Results
5. Conclusion

Table of contents

1. **Background**
2. Problem Formulation
3. Proposed Method
4. Results
5. Conclusion

Feedback for Virtual Reality (VR) Training – VR simulators

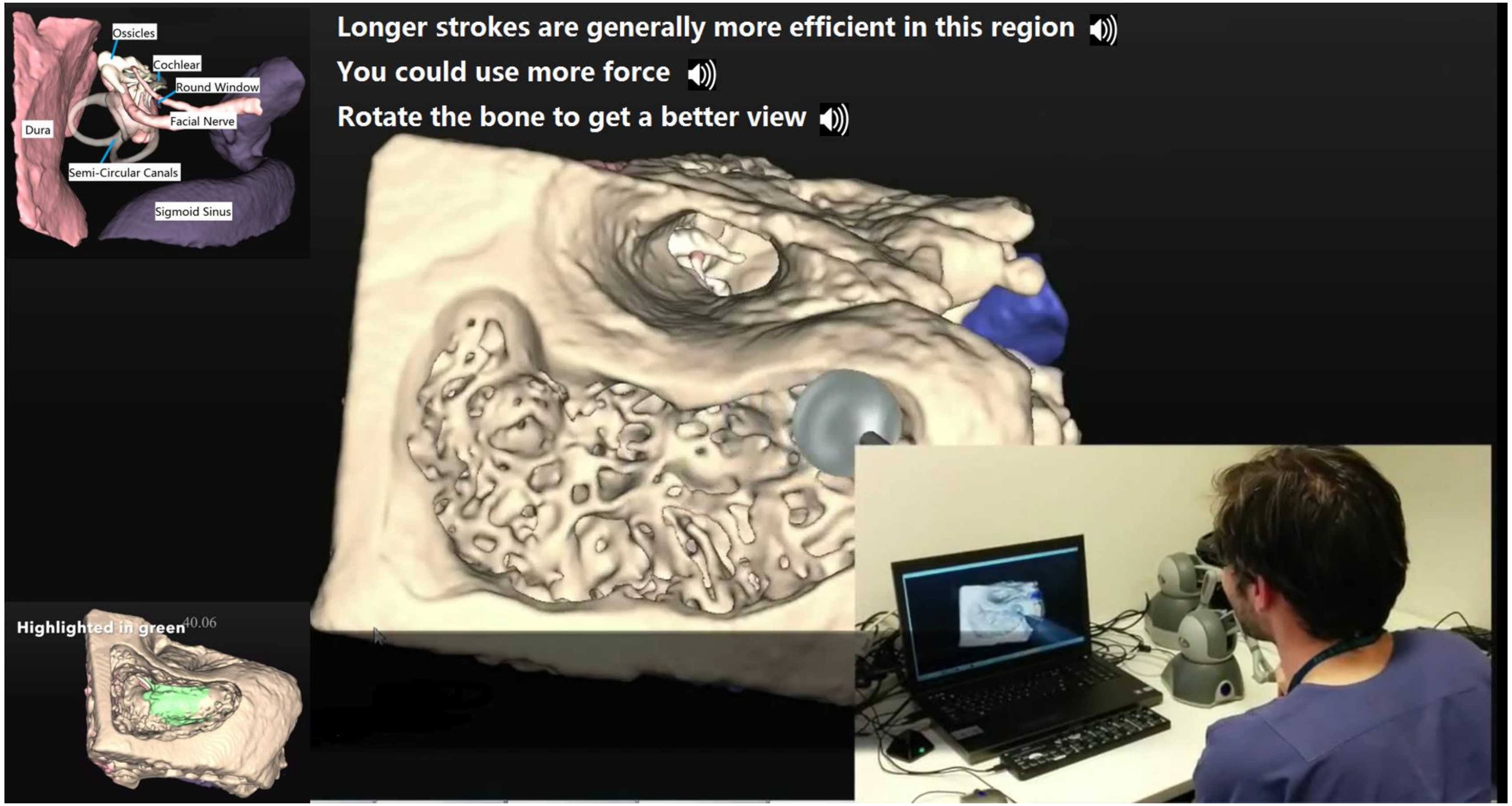
- VR simulators have been widely used in many applications:
  Driver training, surgical training, pilot training, military training, …

A VR driving simulator

A VR surgery simulator

A VR flight simulator

Feedback for VR Training – our simulator



The University of Melbourne Temporal Bone Surgery Simulator

Feedback for VR Training – real-time feedback

VR platforms for training

- Benefits: low-cost, low-risk, convenient accessibility, repeatable practice, etc.
- Drawback: lack of real-time guidance -> needs **automatic real-time feedback** support

❑ Feedback intervention → effective knowledge/skill learning

Traditionally: experts, professors, experienced technician

VR training: it still requires expert's supervision

❑ Benefits of real-time interactive feedback:

✓ Increase learning motivation
✓ Improve performance
✓ Obtain proper skills and correct mistakes
✓ More importantly, improve decision making skills

Feedback for VR Training – real-time feedback

Intelligent tutoring feedback? In real-time?

Is it possible that feedback is generated in real-time automatically and is expert-like?

Challenges:
- Real-time: 1 second after action performed.
- Accurate: correctly identify novice technique.
- Effective: successfully change novice technique to expert technique.
- Simplicity: one feedback only address 1 or 2 aspects of the technique.
- Transferability: similar task with different difficulties or different tasks

Feedback for VR Training – real-time feedback

What is feedback?

High-level:

- Feedback is the ***helpful information*** presented to the trainee about his/her prior behavior which can be used to adjust improve future behaviour.

Low-level:

- ***Actions*** that need to be taken to improve performance.

Feedback for VR Training – real-time feedback

An example of real-time feedback.

Table of contents

1. Background
2. **Problem Formulation**
3. Proposed Method
4. Results
5. Conclusion

Feedback Generation – preliminary

- Skill Vector: how to define user behavior?

Metrics (features): 1) motion-based, 2) time-based, 3) position-based, or 4) system settings
Example:
Drill speed, drill force, trajectory straightness, burr size … in our Temporal Bone Surgical Simulator

- Skill Levels: expert vs novice

Supervised learning:
expert demonstrations → expert; student demonstrations → novice

- Goal: novice skill → expert skill

By changing features in the skill vector.
Example:
Feedback: (**force =0.2**; duration = 0.3) → (**force = 0.5**;duration = 0.3) is **"increase force to 0.5".**

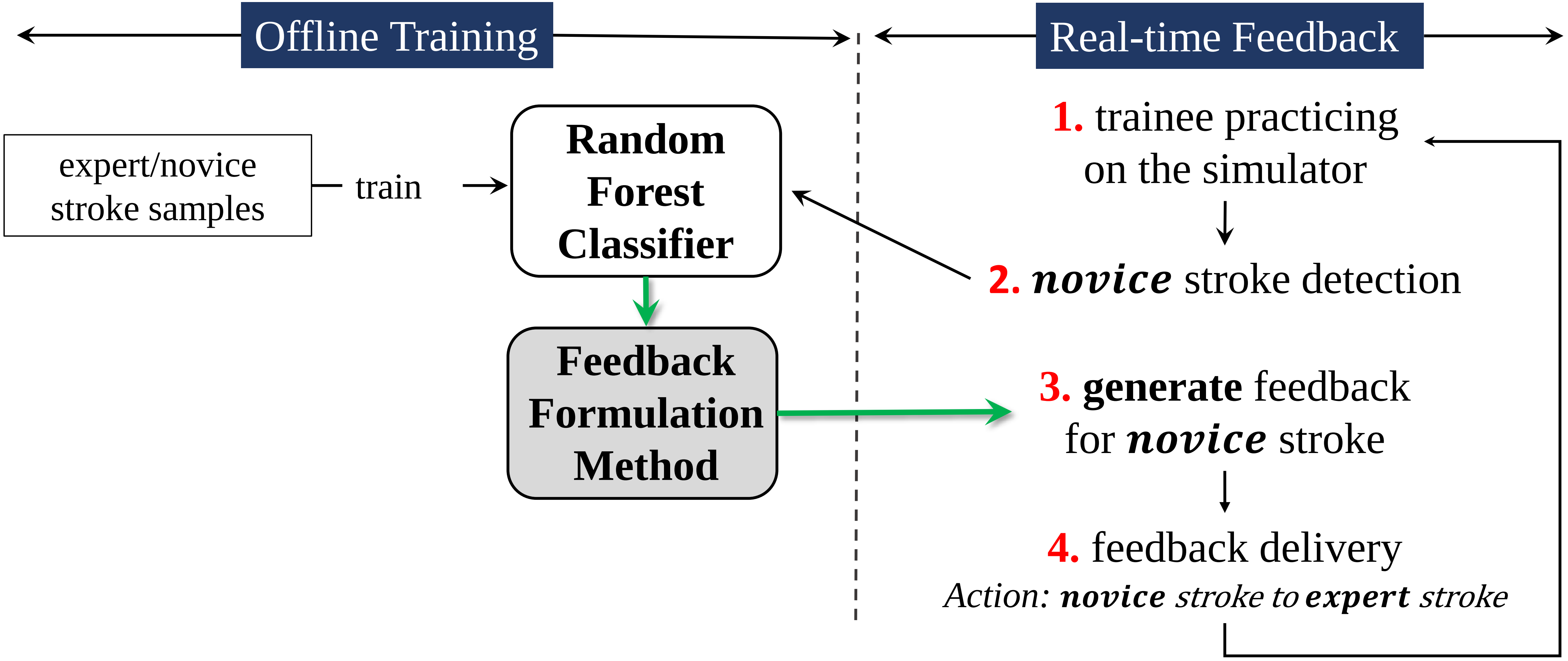
Feedback Generation – preliminary

❑ Skill Vector: how to define user behavior?

Drill speed, drill force, trajectory straightness, burr size … in our Temporal Bone Surgical Simulator

	A	B	C	D	E	F	G	H	I	J	K	L
1	ID	rep	duration	distance	speed	acceleration	straightness	force	endPosX	endPosY	endPosZ	class
2	1	1	1.476112366	0.036040564	0.021341626	0.014457995	0.876401567	1.020398	48	46	191	1
3	1	1	1.785606384	0.034568865	0.017681228	0.009902086	0.917845483	0.890324205	41	42	185	1
4	1	1	0.540367126	0.022240782	0.037401468	0.06921492	0.818044892	0.609978218	58	45	186	1
5	1	1	0.060180664	0.002570428	0.042215855	0.701485358	0.808216952	0.080081559	62	49	190	1
6	1	1	1.180976868	0.016865564	0.011868762	0.010049953	0.800683228	0.904218539	42	44	181	1
7	1	1	0.314849853	0.006115271	0.013830508	0.043927312	0.733340409	0.591966515	47	43	185	1
8	1	1	0.016815186	0.001828157	0.108718163	6.465475062	0.485757881	0.063195315	51	44	187	1
9	1	1	0.611213684	0.024426844	0.037614838	0.061541223	0.946833625	0.930630273	54	41	180	1
10	1	1	1.543510437	0.035317441	0.020468562	0.013261045	0.895981764	1.474808937	58	45	170	1
11	1	1	1.164131164	0.027152068	0.019438321	0.016697707	0.83653916	0.923480384	69	47	193	1
12	1	1	1.063674927	0.011296991	0.009863505	0.009273045	0.87713862	0.817530937	35	41	180	1
13	1	1	1.646224976	0.028158945	0.008764422	0.005323951	0.524442927	1.002791259	47	49	176	1
14	1	1	0.315643311	0.00885854	0.025609742	0.08113507	0.928166771	0.242328232	53	56	168	1
15	1	1	0.560897828	0.008175411	0.012080288	0.021537413	0.802242884	0.622201451	55	60	159	1
16	1	1	0.110610962	0.003005776	0.026695658	0.241347309	0.706379795	0.194428356	74	66	165	1

Feedback Generation – overview



Feedback Generation – existing methods

- Rule-based: *fixed rules, low flexibility*

"follow-me" approach: [Rhienmora et al., 2011]
"step-by-step": [Wijewickrema et al., 2016]

- Pattern-based: *representative patterns, low accuracy/effectiveness*

Time series pattern: [Forestier et al., 2012]
Expert/novice skill pattern: [Zhou et al., 2013a]

- Prediction models: *extract knowledge from trained model, marginal improvements*

Decision tree: [Yang et al., 2003]
Random forests: [Zhou et al., 2013a; Cui et al., 2015]

Feedback Generation – existing methods

Method	**Type**	**Effectiveness**	**Transferability**	**Real-time**
Rhienmora et al., 2011	Rule-based	√	X	√
Wijewickrema et al., 2016	Rule-based	√	X	√
Forestier et al., 2012	Pattern-based	X	√	√
Zhou et al., 2013a	Pattern-based	X	√	√
Yang et al., 2003	Decision tree based	X	√	√
Zhou et al., 2013a	Random forests based	X	√	√
Cui et al., 2015	Random forests based	√	√	X
Our model (NNFB)	**Neural network based**	**√**	**√**	**√**

Feedback Generation – problem definition

Given a prediction model $\mathrm{N}(x)$ and a novice instance x_0, the problem is to find the optimal action $A: x_0 \rightarrow x$ that changes x_0 to an instance x under limited cost C such that x has the highest probability of being in the expert class:

$$\underset{x}{\operatorname{argmax}}\, N(x), \quad \text{subject to} \quad loss(x_0, x) < \mathrm{C}$$

For example, the action $A: (force = 0.2,\ speed = 0.3) \rightarrow (force = 0.5,\ speed = 0.3)$ *translates to the feedback "**increase force to 0.5**".*

- In VR training, the number of feature changes should be kept low to decrease cognitive load and avoid distraction:

$$loss(x_0, x) = \|x_0 - x\|_0$$

- Efficiency: done in 1sec.

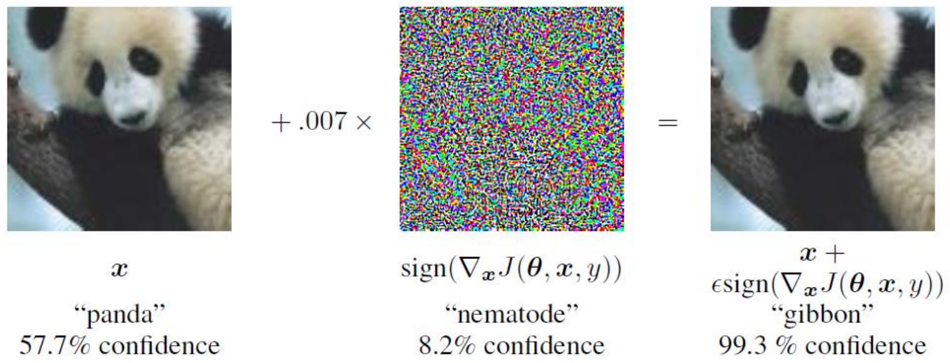
Table of contents

1. Background
2. Problem Formulation
3. **Proposed Method**
4. Results
5. Conclusion

Neural Network based Feedback Generation

Our Contributions:

- ❑ We verified that neural network based **adversarial perturbation** can be a foundation for feedback.



Adversarial examples: imperceptible small carefully-designed noise can fool deep networks. [Szegedy et al., 2013, Goodfellow et al., 2014]

Our method (NNFB)

- The perturbation strategy is the same, but the constraints are different:

✓ larger perturbation (opposite to imperceptible small)

✓ fewer number of feature changes

- ❑ The simplicity of feedback can be done by a L_1 regularization term.
- ❑ Real-world suitability can be done by a bounded adversarial update.

Neural Network based Feedback Generation

- **Step 1:** Pre-train a neural network classifier offline with loss $J_\Theta(x, y)$, via supervised learning.
- **Step 2:** For a real-time novice skill vector x_0 and a target (expert) skill level y^*, adversarially perturb x_0 iteratively:

$$x = x_0$$

$$x = x - \varepsilon S_x\left(xS_x - \frac{\boldsymbol{a}}{2}(1 + S_x) + \frac{\boldsymbol{b}}{2}(1 - S_x)\right)$$

$$S_x = \operatorname{sign}(\nabla_x(J_\Theta(x, y^*) + \lambda \|x - x_0\|_1))$$

- **Step 3:** clip away small changes and generate feedback: $A: x_0 \to x$
- **Step 4:** Deliver *feedback* to trainee in the form of audio instructions in order to enhance performance.

Table of contents

1. Background
2. Problem Formulation
3. Proposed Method
4. **Results**
5. Conclusion

Experimental Setup

- **Datasets:** collected by temporal bone surgery simulator

$\mathcal{D}_1$: cortical mastoidectomy surgery → 60K skill instances (28K expert, 32K novice)

$\mathcal{D}_2$: posterior tympanotomy surgery → 14K skill instances (9K expert, 5K novice)

- **Compared methods:** prediction model based methods

1. Split Voting (**SV**): decision tree based (Zhou et al., 2013a)
2. Integer Linear Programming (**ILP**): random forest based (Cui et al., 2015)
3. Random Iterative (**RI**): iterative approach with random forest (Cui et al., 2015)
4. Random Random (**RR**): random perturbation (baseline)
5. Neural Network based Feedback Generation (**NNFB**): the proposed method.

Experimental Setup

❑ Evaluation metrics:

1. effectiveness: the percentage of successful improved X_0: $effectiveness = \frac{|\{x|N(x)=expert\}|}{|\{x\}|}$

2. time-cost: time (in seconds) on average spent to generate one feedback for a novice instance x_0

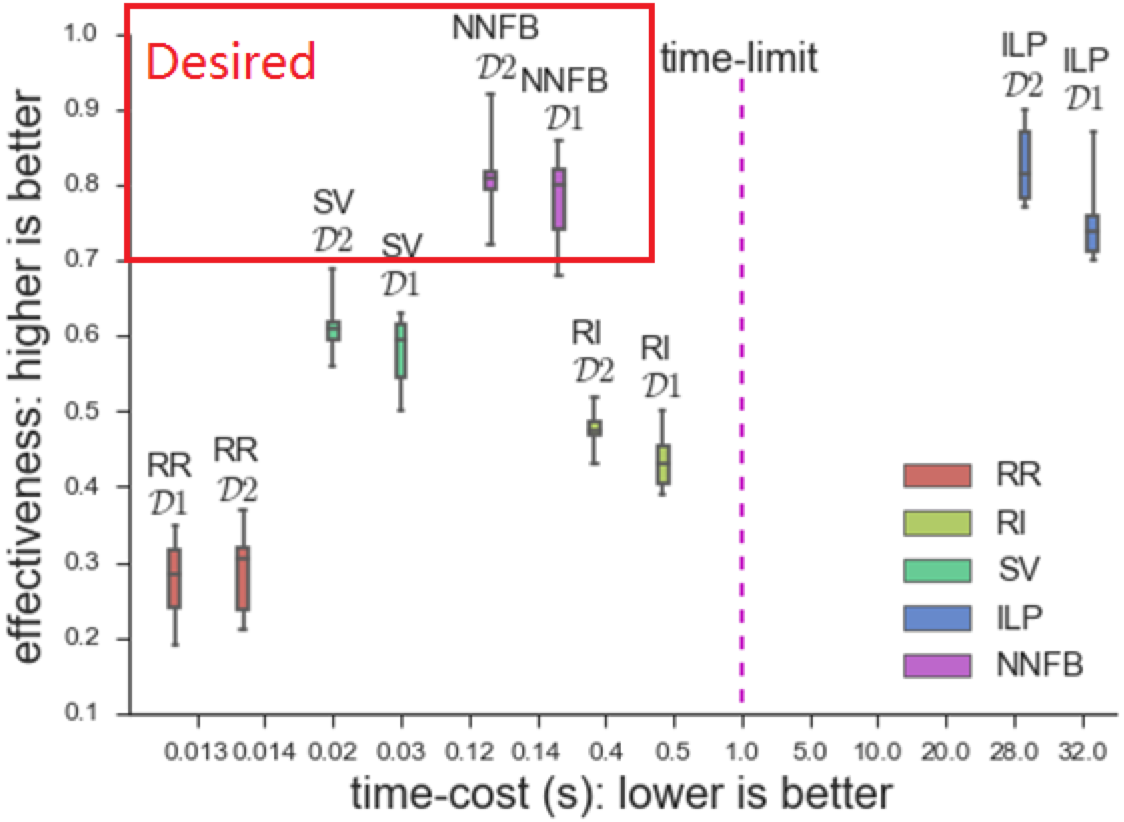
❑ Evaluation classifiers:

evaluation classifiers are pseudo independents experts trained on different data.

Neural Network (NN), Random Forest (RF), Logistic Regression (LR), SVM (RBF kernel), Naive Bayes (NB), KNN (K = 10)

- **Step 1:** Given n novice instances X_0, apply different feedback generation methods to change $x_0 \in \mathrm{X}_0$ to expert instance: x.
- **Step 2:** Evaluate the quality of X: $\{x\}$ using **6 evaluation classifiers**

Experimental Results



- ✓ Higher effectiveness
- ✓ Lower time-cost

Comparison results (effectiveness vs time). $\mathcal{D}_1$ and $\mathcal{D}_2$ are two datasets.
NNFB is the proposed method.

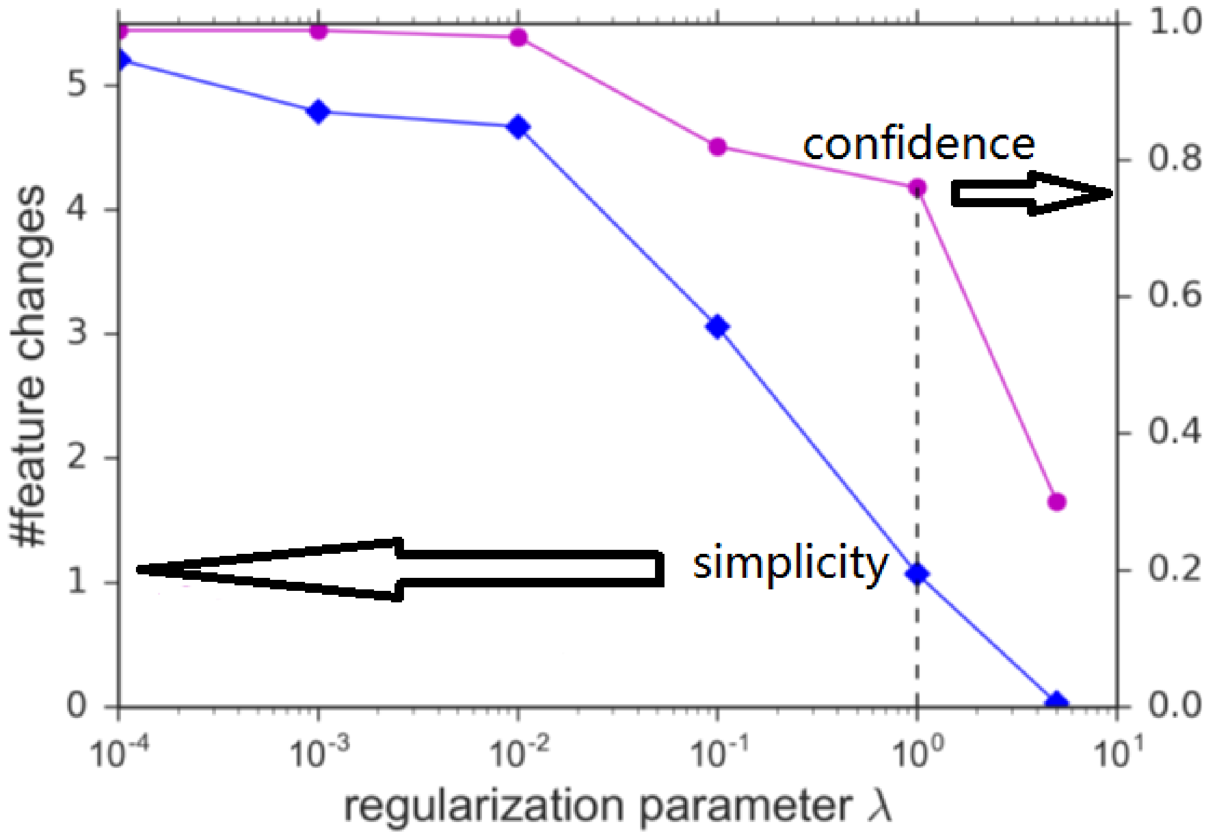
Experimental Results

Table 1: *Comparison of effectiveness (mean±std). Best results are in bold.*

		NN	RF	LR	SVM	NB	KNN
$\mathcal{D}1$	RR	0.19±0.06	0.23±0.10	0.35±0.07	0.27±0.06	0.32±0.12	0.30±0.05
	RI	0.44±0.07	0.39±0.04	0.50±0.08	0.46±0.06	0.42±0.12	0.40±0.08
	SV	0.63±0.07	0.59±0.06	0.60±0.07	0.62±0.06	0.50±0.11	0.53±0.07
	ILP	0.72±0.04	**0.87±0.00**	0.71±0.05	0.76±0.04	**0.70±0.11**	**0.76±0.04**
	NNFB	**0.86±0.01**	0.82±0.08	**0.78±0.05**	**0.82±0.04**	0.68±0.14	0.73±0.08
$\mathcal{D}2$	RR	0.21±0.04	0.22±0.07	0.29±0.04	0.37±0.02	0.32±0.11	0.32±0.06
	RI	0.48±0.04	0.49±0.04	0.47±0.09	0.52±0.05	0.47±0.12	0.43±0.10
	SV	0.61±0.08	0.69±0.04	0.62±0.05	0.61±0.07	0.56±0.11	0.59±0.04
	ILP	0.88±0.04	**0.90±0.02**	0.79±0.07	**0.77±0.03**	0.78±0.12	**0.84±0.09**
	NNFB	**0.92±0.02**	0.82±0.06	**0.81±0.07**	0.72±0.05	**0.79±0.11**	0.81±0.07

NNFB achieved comparable performance to ILP and outperformed others methods across all evaluation classifiers.

Experimental Results



Increase L_1 regularization parameter λ yields simple but confident feedback.

Table of contents

1. Background
2. Problem Formulation
3. Proposed Method
4. Results
5. **Conclusion**

Conclusions

- Adversarial perturbation with neural networks can be used to generate confident feedback efficiently.
- The proposed **NNFB** method is general more effective than existing feedback generation methods while remains low time-cost.
- The proposed L_1 regularization perturbation generates simple yet confident feedback.

References

[Rhienmora et al., 2011] Phattanapon Rhienmora, Peter Haddawy, Siriwan Suebnukarn, and Matthew N Dailey. Intelligent dental training simulator with objective skill assessment and feedback. Artificial intelligence in medicine, 52(2):115–121, 2011.

[Zhou et al., 2013a] Yun Zhou, James Bailey, Ioanna Ioannou, Sudanthi Wijewickrema, Gregor Kennedy, and Stephen OLeary. Constructive real time feedback for a temporal bone simulator. In MICCAI, pages 315-22. 2013.

[Zhou et al., 2013b] Yun Zhou, James Bailey, Ioanna Ioannou, Sudanthi Wijewickrema, Stephen O'Leary, and Gregor Kennedy. Pattern-based real-time feedback for a temporal bone simulator. In VRST, pages 7–16, 2013.

[Yang et al., 2003] Qiang Yang, Jie Yin, Charles X Ling, and Tielin Chen. Postprocessing decision trees to extract actionable knowledge. In ICDM, pages 685–688, 2003.

[Cui et al., 2015] Zhicheng Cui,Wenlin Chen, Yujie He, and Yixin Chen. Optimal action extraction for random forests and boosted trees. In KDD, pages 179–188, 2015.

[Szegedy et al., 2013] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. arXiv:1312.6199, 2013.

Q&A