A Method to Assess and Argue for Practical Significance in Software Engineering

A Method to Assess and Argue for Practical Significance in Software Engineering

Richard Torkar, Carlo A. Furia, Robert Feldt, Francisco Gomes de Oliveira Neto, Lucas Gren, Per Lenberg, and Neil A. Ernst

Abstract—A key goal of empirical research in software engineering is to assess practical significance, which answers whether the observed effects of some compared treatments show a relevant difference in practice in realistic scenarios. Even though plenty of standard techniques exist to assess statistical significance, connecting it to practical significance is not straightforward or routinely done; indeed, only a few empirical studies in software engineering assess practical significance in a principled and systematic way.
In this paper, we argue that Bayesian data analysis provides suitable tools to assess practical significance rigorously. We demonstrate our claims in a case study comparing different test techniques. The case study's data was previously analyzed (Afzal et al., 2015) using standard techniques focusing on statistical significance. Here, we build a multilevel model of the same data, which we fit and validate using Bayesian techniques. Our method is to apply cumulative prospect theory on top of the statistical model to quantitatively connect our statistical analysis output to a practically meaningful context. This is then the basis both for assessing and arguing for practical significance.
Our study demonstrates that Bayesian analysis provides a technically rigorous yet practical framework for empirical software engineering. A substantial side effect is that any uncertainty in the underlying data will be propagated through the statistical model, and its effects on practical significance are made clear.
Thus, in combination with cumulative prospect theory, Bayesian analysis supports seamlessly assessing practical significance in an empirical software engineering context, thus potentially clarifying and extending the relevance of research for practitioners.

Index Terms—practical significance, statistical significance, Bayesian analysis, empirical software engineering

1 Introduction

A main goal of research in empirical software engineering (ESE) is assessing practical significance: what is the impact of the research findings in realistic scenarios? To this end, statistical analysis has been used extensively in ESE for decades. Nonetheless, the bulk of research has focused on defining and implementing guidelines for experimental design (from case studies [1] to grounded theory [2] and experiments [3]) and statistical analysis (from statistical testing [4] to Bayesian modeling [5]). In contrast, practical significance is rarely discussed explicitly or quantitatively [6].

The most common approach to assessing the significance of findings is built on top of statistical significance, which is extended in a quantitative way. A common example are effect size measures (such as Cohen's d, or the size of coefficients in a regression model): if the effect size of a technique A is markedly bigger than the one of another technique B, this is taken as an indication that A performs better than B in practice. This common approach overlooks the issue that assessing practical significance on statistical measures such as effect sizes makes it hard to ensure that the statistics accurately reflect expert knowledge. In particular, practitioners (who are the experts) may not be familiar with the nuances of the various statistical techniques and how they are used.

Furthermore, showing statistically that one technique performs better than another one does not automatically mean that this makes a difference in practice. Using effect sizes frames practical significance as a general property [7]; however, it is much more likely to be a context-dependent property, as whether a technique will be better than another in practice depends on the context where those techniques will be deployed. Therefore, a quantitative assessment of practical significance should be expressible in terms of (or, at least, clearly connected to) measures in the application domain that are used by the domain experts. For example, return on investment, time, and personnel costs are all measures that are appropriate for an evaluation of economic impact.

Grounding practical significance in domain-specific metrics also supports a clear communication of the expected impact of the solutions that have been empirically assessed; in other words, how each solution would help practitioners [8], and how they could choose the one most appropriate for their needs [9], [10]. Establishing such a clear communication would ultimately enhance research's long-term impact [9], [11].

In this paper, we demonstrate how a combination of Bayesian analysis [5], [12], [13] and cumulative prospect theory [14], [15] provides a serviceable means of assessing practical significance in ESE. Using data from a case study of testing practices [16] (comparing exploratory testing to testing based on predefined test cases), we illustrate how

R. Torkar, R. Feldt, L. Gren, P. Lenberg, and F. Gomes de Oliveira Neto are with the Department of Computer Science and Engineering, Chalmers and University of Gothenburg, Sweden. e-mail: torkarr@chalmers.se.
R. Torkar is also with the Stellenbosch Institute for Advanced Study (STIAS), Wallenberg Research Centre at Stellenbosch University, Stellenbosch, South Africa.
C. A. Furia is with the Software Institute of USI Università della Svizzera italiana, Via Giuseppe Buffi 13, I-6904 Lugano, Switzerland.
L. Gren is also with Blekinge Institute of Technology, Karlskrona, Sweden.
N. A. Ernst is with the Department of Computer Science, University of Victoria, 3800 Finnerty Rd, Victoria, BC V8P 5C2, Canada

one can formulate and assess different measures of practical significance, all expressed in terms of metrics that make sense in the application domain (such as the hourly cost and seniority of programmers).

We use Bayesian statistics to design and fit a model of the empirical data. The model is quantitative, incorporates expert knowledge in the domain metrics of interest, and can be used to perform predictions. We then use cumulative prospect theory to connect probabilities in the Bayesian model to utility metrics of possible outcomes. The connection is actionable, in that it supports decision making based on the applicability, risks, and costs of different scenarios. If one would take this one step further, these costs per scenario could greatly reduce the need for human subjects for validating research results as cumulative prospect theory will provide us with strong indications of what decision a human will take.

1.1 Cumulative Prospect Theory

Cumulative prospect theory (CPT) is a framework developed in behavioral economics to model decisions under risk and uncertainty [14] and has been applied for practical decision making, e.g. in medicine [17]. CPT models several aspects of human decision making, such as sensitivity to how options are framed, non-linear sensitivity to risk, and loss aversion; based on these factors, a CPT model defines the utility of a certain decision's outcome.

Software engineering practitioners are faced with decision making under uncertainty, so CPT is also a useful framework in this domain. As a simple example, consider a manager who is organizing a code review. Two options are available to them: approach A, which guarantees a value of \$940; and approach B, which gives a value of \$1000 with 95% probability, and no value (\$0) with 5% probability. CPT indicates that most managers will choose A given its certainty, even though the expected utility of B is slightly better (as $0.95 \times 1000 = \$950 > \$940$). This behavior is a manifestation of risk aversion, which may also depend on how the problem is framed.

CPT can provide a suitable model of how decisions are made in software engineering practice as well. As we show in this paper, empirical data can be used to fit probabilistic models of different outcomes; the probabilities correspond to risks in a CPT model. The latter also acts as a sort of "high-level interface" for the practitioners and decision makers, who do not have to understand the statistical model but can reason in the more familiar terms of risks and utility values of each possible outcome. Domain expertise remains crucial to build a suitable CPT model: in the previous example of code review approaches A and B, domain experts would estimate the profits and costs associated with each option.

1.2 Proposed Solution

We combine cumulative prospect theory and Bayesian statistics to assess practical significance of ESE data. Figure 1 gives an overview of our approach. We use CPT to model the possible decisions, how they are framed, and the risks associated with them. Then, we use Bayesian statistical analysis to infer the probabilities that quantify the risk of each decision. While in principle any statistical approach could work, Bayesian models are better suited because they provide a detailed *posterior* distribution of the possible outcomes (instead of just point or interval estimates), which can be seamlessly combined with CPT models. In addition, Bayesian models are easy to interpret [5] and naturally incorporate expert knowledge and assumptions through the use of *priors*.

Our approach is applicable to many ESE topics and subject areas, as long as the investigation is suitable for a quantitative analysis based on statistical methods. In this paper, we do not investigate the connection between our proposed framework and qualitative analysis in empirical software engineering, as both approaches are complementary. In other words, we assume that there are strategies to identify and measure values that can be translated into decisions framed as weighting functions.

As an illustration (Figure 2), consider that we investigate whether to favour the execution of a bigger, costlier, but more thorough test suite versus smaller, cheaper, and relatively superficial test suites. Using data collected from mining software repositories and an expert's opinion, we can create a generalized linear model (GLM) to analyse how the different sizes of test suites affect executions costs, as well as the number of failures that are revealed and must be fixed. Ultimately, the model provides input to calculate the utility of both types of test suites in connection to different probable scenarios, such as the test execution costs based on the unlikely situation of revealing too many failures.

We evaluate the feasibility of this approach in three steps. First, we reanalyze a previously published empirical study [16] using Bayesian techniques. Second, we combine the Bayesian model with CPT under plausible practical application scenarios. Third, we ask practitioners to compare the information provided by the original study [16], to that derived by our combination of Bayesian statistics and CPT, and to indicate which better supports their decisions. Overall, we demonstrate how to build a rigorous model of practical significance, and the advantages of reporting significance results in a way that is grounded in concrete decision-making scenarios—in contrast to the traditional approach that presents general statistics in a more abstract form.

In summary, the contributions of this paper are:

- An approach that connects statistical inference and practical significance to frame empirical findings in a way that supports practitioners in making decisions.
- A case study of applying cumulative prospect theory in ESE.
- A reanalyses of previously published data [16], which revisits the original findings and extends them to a context of practical significance. Our analysis is reproducible and available online [1].

This paper is structured as follows. Section 2 presents the background for both Bayesian analysis and CPT, so as to introduce readers to the essential terminology and steps of both techniques. We present our analysis in Section 3, followed by the application of CPT in Section 4. The results from the validation of our approach with practitioners are

1. https://github.com/torkar/docker-b3

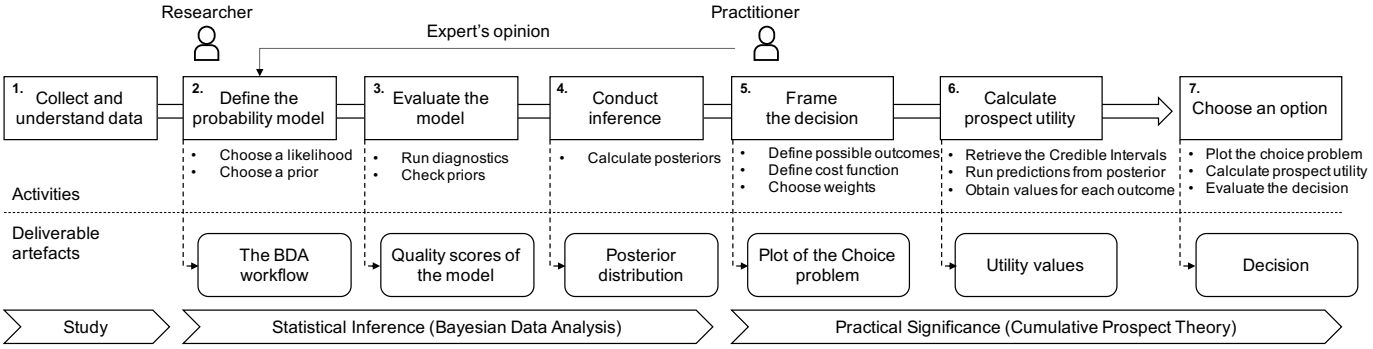

Fig. 1. Assessing practical significance using a combination of Bayesian analysis and cumulative prospect theory.

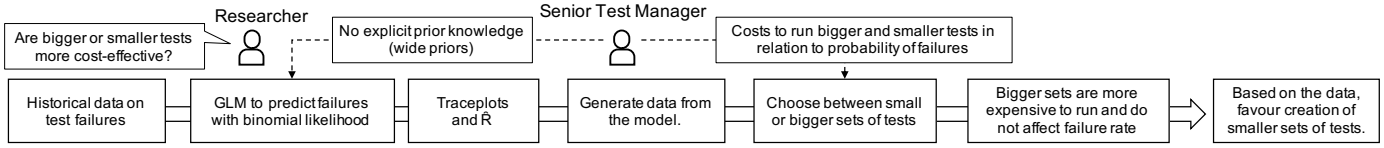

Fig. 2. An illustration of our approach using BDA and CPT applied to the decision between bigger or smaller test suites for cost-effective testing. The GLM enables the prediction of failures based on the test suite sizes via a Binomial likelihood. The expert (senior test manager) provides the cost model of running a test suite (e.g., including costs for both size and fault fixing) that is then framed as choices between test suite sizes with corresponding utility values. Practitioners then choose the outcome with better prospect utility.

presented in Section 5. In Section 6 we discuss the implications of our work, while in Sections 7 and 8 we present threats to validity and conclude the paper.

2 Background and Related Work

In order to present the background of our research, we introduce the essential terminology and techniques of Bayesian statistical analysis and CPT. We also discuss related work in ESE literature that also targets the practical impact of research findings.

2.1 Bayesian Analysis

Data analysis relies on statistical analysis to infer properties of a population that follows an underlying probability distribution. Since the actual underlying distribution is often unknown, statistical inference estimates probabilities by generalizing the frequencies observed in finite *samples* of the population.

There exist different families of probability distributions, such as uniform, normal, binomial, Poisson, beta, etc. Each distribution in a certain family is characterized by the values of one or more parameters $\theta_1, \ldots, \theta_n$, which fix the distribution's shape. For instance, a normal (also called Gaussian) distribution has two parameters: mean μ and standard deviation σ. The key goal of statistical analysis is to estimate a distribution parameter θ from *sampled data* D that was drawn from the population. In Bayesian statistics, this is expressed as estimating the probability $P(\theta \mid D)$ of the parameters given the data, which obeys Bayes' theorem:

$$P(\theta \mid D) = \frac{P(D \mid \theta) \times P(\theta)}{P(D)}, \quad \text{where:} \tag{1}$$

- $P(\theta \mid D)$ is the *posterior distribution* and is what we want to estimate.
- $P(D \mid \theta)$ is the *likelihood* that the data was drawn from a distribution with parameters θ.
- $P(\theta)$ is the *prior*, which encodes prior knowledge by constraining plausible values for the parameters θ independent of the data.
- $P(D)$ is simply a normalizing constant.

As a concrete example, imagine that we are investigating the number of tests failing during a build of a certain project.[2] We would like to infer the failure rate, i.e., the probability p_i of a test failing in an arbitrary build. The binomial distribution family represents sequences of n events each with a fixed probability of success p. In our example, the n events are the tests that are executed in each build, and p_i is the probability of failure for test i. Thus, draws from the distribution Binomial(n, p) capture the likelihood of observing a certain number of tests failing given parameters n and p.

The case study in Section 3 demonstrates these concepts in greater detail. For additional technical details about how Bayesian analysis is applied, we refer to [5] and [12]. The key features of a Bayesian approach, which makes it best suited to analyze practical significance are:

1) It can incorporate prior knowledge.
2) It produces a posterior predictive distribution (PPD).
   a) The PPD is conditioned on the observed data.
   b) The conditioning allows us to update our beliefs of the unknowns.
   c) The variation indicates any remaining uncertainty in our beliefs about the unknowns.
   d) It produces estimates of the standard deviation of the marginal posterior distributions.

2. For the sake of simplicity, we assume that a test either passes or fails (no flaky behavior or timeouts are possible).

3) It produces Bayesian uncertainty intervals, which quantitatively measure degree of belief after seeing the data.

2.2 Cumulative Prospect Theory

A fitted Bayesian model can be used to *simulate* new scenarios that generalize those in the observed data. Each scenario is associated with different outcomes, costs, and potential benefits. In the example of testing outlined in the previous section, different testing practices lead to different failure rates and have different application costs (for example because they require more developers); on the other hand, failing tests have to be fixed, which introduces additional costs. We use cumulative prospect theory (CPT) [14] to support a decision-making process, which is based on scenarios that are modeled statistically.

CPT is a widely used decision-making modeling framework, which accounts for experimentally demonstrated features of human behavior when making a decision:

- *Loss aversion* is the preference for avoiding losses over gaining advantages—even when the latter would outweigh the former.
- *Framing effects* refer to the widespread psychological phenomenon that the same data may lead to very different choices according to *how* (in which scenario) the data is presented.
- *Nonlinear preferences:* humans associate risk to the 0–100% probability scale in a way that is not uniform. For example, the difference between a 99% and 100% risk is considered more significant than the difference between a 10% and 11% risk.
- People tend to be *risk averse*, except in specific scenarios where they are more likely to actively *seek risks*: when there is a small probability of winning a large prize and when choosing between a sure loss and a substantial probability of an even larger loss.
- *Source dependence* refers to the influence of domain expertise on preferences. Decision makers usually feel more confident within their area of expertise even when they have limited and noisy data—often even more confident than when they have detailed data about an area they are not familiar with [18].

Concretely, CPT represents a decision using a function $\nu : X \to \mathbb{R}$ that maps each possible outcome $x \in X$ of the decision to its value $\nu(x)$. A positive value is a gain, and a negative one is a loss. We associate each outcome x with a weight that reflects the outcome's probability corrected to account for how it is subjectively perceived (according to the phenomena listed above). More precisely, CPT provides standard weighting functions that can be applied to any probability distributions of outcomes to derive the "subjective" probability associated with any outcome. The subjective expected utility value $\mathbb{E}_U(X)$ of the decision is the weighted average of the value of all possible outcomes—each weighted by its subjective probability, which in turn is based on the outcome's "objective" probability (from a statistical model).

Continuing the example of the failing tests, we imagine a manager choosing between two testing approaches A and B. Each approach leads to a series of outcomes with different probabilities. The probabilities come from a statistical analysis of the available data from using A and B in the past. For instance, we may learn that approach A: 1) fails to produce bug-revealing tests in 3% of its applications; 2) generates too many redundant failing tests in another 3% of applications; 3) and works fairly successfully in the remaining applications. According to CPT, each outcome's probability is weighted so as to reflect loss aversion or other subjective phenomena. For instance, if the losses associated with outcomes 1 and 2 above are very large, the overall expected value of testing approach A should be small because the person in charge of the decision is more likely to avoid any risk of large losses.

An additional advantage of using CPT on top of a pure probabilistic model is that the notion of subjective expected value of a decision is intuitive and understandable by decision makers—building a kind of abstraction layer on top of a harder-to-interpret probabilistic model.

2.3 Practical Significance in ESE

Practical significance is important in an engineering discipline where research should inform "solutions to practical problems" [19].[3] Indeed, the emergence of empirical software engineering was driven in part by the desire to identify effectiveness and significance in practice. For example, one of the early guides to ESE research [20] highlights the need to "differentiate between practical and statistical significance".

Despite this early agenda, practical significance is rarely explicitly discussed in software engineering research publications [6]. When it is, there is a tendency to conflate it with statistical measures of strength of evidence. Unfortunately, even when rigorous statistics are applied to carefully controlled experiments, they may not generalize to realistic conditions [21]–[23].

Other kinds of evidence (besides controlled experiments) may also support claims of practical significance. For example, case studies support generalization by providing instances that probe the boundaries of the applicability domain [24]. Another approach is framing software engineering as a design science [8], [25], which focuses on the feedback loop between problem space and solution space. Then, practical significance can be tested when a solution is deployed in a practical setting.

Another important aspect of generalizability, and hence practical significance, that can be evaluated empirically is *scalability* to realistic settings [25]. Even more fundamental is that the investigated *problem* must be important to industry practitioners. This is tricky because the practitioner's view is typically tied to a specific problem context. For example, [7] mentions the example of a defect prediction study that focuses on predicting defects. Most of the value for practitioners does not lie in defect prediction *per se* but rather in how this information can guide decisions and explain the origin of defects. In this paper, we use CPT to model such connections between a technique's raw performance and its value in terms of decision making.

Effect sizes and replications are the main tools of statistical analysis that can support assessing practical significance. Effect size measures connect the outcomes of statistical tests

3. Still, one's definition of "practical" may differ from another's.

to a measure of real-world impact. There are numerous effect size approaches, including R^2 in regression models (portion of variance explained), regression coefficients, Cohen's d (standardized mean difference), Hedge's g, Cliff's δ (differences for ordinal data), and odds-ratios to capture relative effects.[4]

When researchers translate a raw effect size number into practical terms (moving from the estimate to population effects), they often use Cohen's t-shirt sizing approach (S,M,L). Correll [26] describes in detail the problems with this categorical approach to effect size, including inconsistent bin thresholds, and Cohen himself said binning was a last resort: "*contextual*, subjective judgment of observed effect sizes must be made and a ritualized interpretation avoided (emphasis ours, as quoted in [27])". The effect size is important, because frequentist analysis uses estimates of expected effect size to determine the appropriate sample size, given a particular power threshold (typically 80%). Notwithstanding these existing and well-investigated effect size metrics, in empirical software engineering research reporting effect sizes is not widespread [27], albeit, as can be seen from [6], a positive trend is visible (when not controlling for the possibility that journals publish more papers on a yearly basis). More importantly, effect size ignores the context of decision making. A raw number reflecting (for example) the standardized difference of means is hard for practitioners to interpret and must be contextualized. Decision analysis is the use of results from statistical inference to support decision-making in a specific context, for which effect size or, as in this paper, posterior predictive distributions, serve as inputs.

3 Connecting Statistical Significance with Practical Significance

Statistical significance in itself does not imply that an effect has practical value or utility. Rather, practical significance is assessed on top of a statistical model that summarizes the data and its variability. As a concrete case, to show how such an analysis can be performed, we first analyze a previously published study on testing practices [16], following the first four steps in Fig. 1. Below, we provide further details on this case study, its statistical modeling and significance, as well as noting differences between different statistical approaches.

3.1 The Case Study

The study described in [16] compares exploratory testing and testing based on documented test cases. In exploratory testing, developers are free to experiment with the system in an interactive fashion. In test-case based testing, developers are required to document their work by writing test cases and oracles, which can be re-executed at any later time.

The experiment we consider here involved 35 developers, classified into two categories according to their experience (years spent as software developers): 12 less experienced testers, and 23 more experienced ones. The developers included both industrial practitioners and software engineering students. Unsurprisingly, on average, members of the former group had more experience than members of the latter; but some students were still classified as "more experienced" and some practitioners as "less experienced".

4. See https://rpsychologist.com/cohend/ for an interesting visual tool for exploring effect sizes using Cohen's d.

TABLE 1
Summary statistics about the experimental data. For each category of developers (low or high): the number of developers, and the median, mean, standard deviation, minimum, and maximum number of faults each of them found during the experiments.

		faults found through testing				
experience n		median	mean	sd	min	max
low	12	3	4.3	4.3	0	20
high	23	5	6.0	5.3	0	18
any	35	4	4.9	4.7	0	20

Following a 2×2 cross-over experimental design, each developer was randomly assigned to one of the two testing approaches (exploratory or test-case based), so that each combination of approach and experience included a similar number of developers. After the first session the developers switched techniques.

During the experiment, developers had to apply their assigned testing approach to find as many faults as possible when testing an integrated development environment (`jEdit`). The number of faults found during the allotted time (two 90-minute sessions) measured each developer's *effectiveness*—which should reflect, thanks to randomization and experimental design, the intrinsic effectiveness of different testing techniques. Table 1 summarizes the experimental data.

3.2 Statistical Modeling

The main goal of our statistical analysis is inferring a probability distribution of the number of faults detected by developers testing the system. Thus, the outcome is a natural number ($\mathbb{N}$) *faults*, which depends on two categorical predictors capturing the testing *approach* (exploratory or test-case based) and the developer's *experience* (low or high) in each trial.

Population-level effects model. Let us first consider a general linear model $\mathcal{M}_1$. Since *faults* is a non-negative integer variable representing a count, a Poisson distribution is a suitable *likelihood* distribution [12] relating predictors and outcome, as shown in Eq. (2) in $\mathcal{M}_1$'s definition below. Both *faults* and λ have a subscript i, which makes it explicit that we evaluate the model for each *subject* i among all 35 developers (the dependence on the subject is usually left implicit; we make it explicit so that it is clear what depends on the subject and what does not). The Poisson distribution's rate λ is the log-linear function (Eq. (3)) of predictors *approach* and *experience*—each modeled as a binary indicator variable for the two possible approaches and experience levels. Finally, to apply Bayes' theorem we need to define *priors* for $\mathcal{M}_1$'s parameters α, β_a, and β_e. A standard choice, which works well in most cases, is a weakly-informative prior such as a normal distribution with zero mean and moderate standard deviation, as shown in Eq. (4). This prior does not bias the effect that the predictors may have towards positive or negative values, and it still allows for a large

range of possible parameter values—even though extreme values (corresponding to very large effects) are increasingly unlikely.[5] Here is the overall definition of $\mathcal{M}_1$.

$$\text{faults}_i \sim \text{Poisson}(\lambda_i) \quad (2)$$
$$\log(\lambda_i) = \alpha + \beta_a \cdot \text{approach}_i + \beta_e \cdot \text{experience}_i \quad (3)$$
$$\alpha, \beta_a, \beta_e \sim \text{Normal}(0, 1.5) \quad (4)$$

Fitting $\mathcal{M}_1$ using the data gives a joint probability distribution on the parameters α, β_a, β_e, which together identify a *posterior* probability distribution of *faults* given the data. According to Eq. (3), the parameters connecting predictors to outcome are the same for every subject in the experiment (that is, they do not depend on i); thus $\mathcal{M}_1$ is a *population-level effects* model.[6]

Varying effects model. Bayesian analysis stresses the importance of modeling data under different assumptions. Therefore, let us consider ways of extending $\mathcal{M}_1$ into a finer-grained $\mathcal{M}_2$, which may capture additional characteristics of the data under analysis.

By looking into the data more closely, we note that over 18% of the developers found no faults during the experiments; that is, outcome *faults* $= 0$ occurs more frequently than what a Poisson distribution predicts. To account for this, we use a *zero-inflated* Poisson distribution as likelihood in $\mathcal{M}_2$. As shown in Eq. (5), such a distribution depends on a rate λ, like in a regular Poisson, but it may produce a count of zero with probability p in each draw. However, unlike λ, for parameter p we use a logit function. As shown in Eq. (8), we assume that only variable *approach* may affect p since all cases of developers finding zero faults occurred when using test-case based testing.

The other modeling assumption of $\mathcal{M}_1$ that we want to reconsider are the population-level effects. To account for the possibility that each subject i may have different intrinsic skills at finding faults, we add an intercept term $\alpha_{\text{SUBJECTS}_i}$ to our linear regression (Eq. (7)). This term represents the "baseline" contribution of each subject to the number of faults that are detected. In summary, this makes $\mathcal{M}_2$ a *varying effects* model.[7]

As in $\mathcal{M}_1$, $\mathcal{M}_2$'s priors are weakly informative; the standard deviation σ_s, for the subject-specific intercept $\alpha_{\text{SUBJECTS}_i}$, follows a half Cauchy distribution (Eq. (9)), which is a common choice [28] for a standard deviation (a non-negative real value). Here is the overall definition of $\mathcal{M}_2$:

$$\text{faults}_i \sim \text{ZIPoisson}(p_i, \lambda_i) \quad (5)$$
$$\log(\lambda_i) = \alpha + \beta_a \cdot \text{approach}_i + \beta_e \cdot \text{experience}_i \quad (6)$$
$$+ \alpha_{\text{SUBJECTS}_i} \quad (7)$$
$$\log(p_i) = \alpha_p + \beta_p \cdot \text{approach}_i \quad (8)$$
$$\alpha_{\text{SUBJECTS}} \sim \text{Normal}(\mu_s, \sigma_s) \quad (9)$$
$$\alpha, \beta_a, \beta_e \sim \text{Normal}(0, 1.5) \quad (10)$$
$$\alpha_p, \beta_p, \mu_s \sim \text{Normal}(0, 1.5) \quad (11)$$
$$\sigma_s \sim \text{Cauchy}^+(0, 1) \quad (12)$$

As for $\mathcal{M}_1$, fitting $\mathcal{M}_2$ using the data gives a joint probability distribution on the parameters $\alpha, \beta_a, \beta_e, \alpha_{\text{SUBJECTS}}, \alpha_p, \beta_p$, which together identify a *posterior* probability distribution of *faults* given the data.

More precisely, the posterior is derived using statistical frameworks such as Stan [29], which work by sampling numerical approximations. Therefore, in practice, the posterior is not an analytical expression but rather a computational object.

Model comparison. We introduced $\mathcal{M}_2$ as a refinement of $\mathcal{M}_1$ based on some features of the data under analysis. This should make $\mathcal{M}_2$ fit the data better, but it may also increase the risk of *overfitting*. Fortunately, Bayesian analysis offers techniques to quantitatively compare models selecting those that achieve the "best" trade-off between fitting the data accurately while avoiding overfitting.[8] To this end, we use an information criterion. An information criterion is a relative measure of how well a model performs out-of-sample predictions compared to other competing models.

In our case, the PSIS-LOO state-of-the-art information criterion [30] indicates that $\mathcal{M}_2$ outperforms $\mathcal{M}_1$ in out-of-sample prediction (see the replication package for a thorough explanation of this part of the analysis). Therefore, we use $\mathcal{M}_2$ in the rest of our analysis, since it captures trends in the data better, while avoiding overfitting.

3.3 Statistical Significance

The posterior is a probability distribution over the parameter space, which quantifies the degree of belief in each possible combination of parameter values. Therefore, it captures probabilistic features of the process we are analyzing, namely how fault detection is affected by the chosen testing approach and the developer's experience.

From the posterior's joint probability distribution we can compute the *marginals* for the parameters of interest. For example, Fig. 3 displays the marginals of coefficients β_a and β_e. Parameter β_a models the effects of the testing *approach* used by each developer (named 'Technique' in the figure)[9]. Since β_a is estimated to be very clearly negative, it means that the chosen testing approach consistently correlates with the number of faults that are found. Since *approach* is a binary variable, with 0 corresponding to exploratory testing and 1 corresponding to test-case based testing, a negative coefficient β_a means that exploratory testing is associated

5. Choosing even weaker priors, such as completely flat ones, would not affect the overall inference but may make sampling less efficient.

6. Population-level effects are often known as "fixed" effects.

7. Varying effects are often known as "random" effects.

8. Posterior predictive checks were conducted for each model to judge the degree of fit.

9. Note that we use 94% rather than 95% intervals in the figure since the former is customary in the field of CPT.

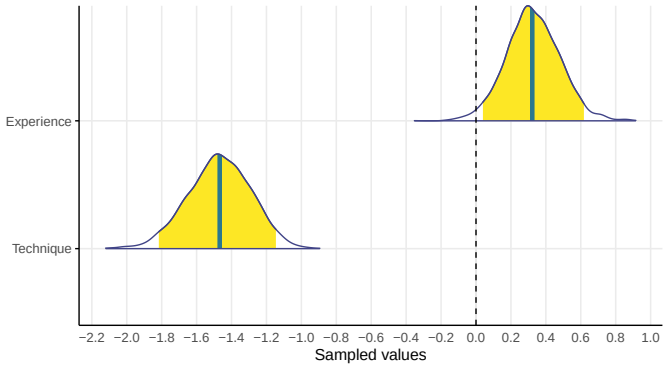

Fig. 3. Posterior marginal probability distributions of β_e (*experience*, top) and β_a (*approach*, bottom named 'Technique'). The thick lines mark the medians, and the yellow areas cover 94% of probability.

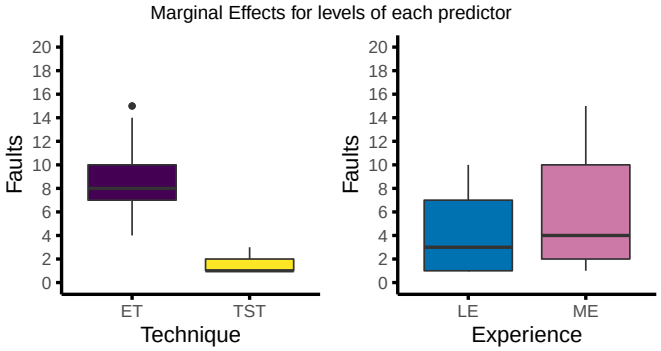

Fig. 4. Left: expected number of detected faults with 94% probability for developers using exploratory testing and test-case based testing. Right: expected number of detected faults with 94% probability for developers with low experience and high experience.

TABLE 2
Summary statistics of all population-level parameters in $\mathcal{M}_2$: mean (estimate), standard deviation (error), and lower and upper endpoint of the 94% probability interval (credibility interval) of the parameter's posterior.

parameter	mean	std. dev.	94% CI	
α	1.95	0.10	1.75,	2.13
β_a	-1.47	0.18	-1.83,	-1.13
β_e	0.33	0.15	0.03,	0.63
α_p	-4.61	1.35	-7.75,	-2.56
β_p	3.39	1.61	0.56,	6.80
σ_s	0.29	0.09	0.10,	0.45

TABLE 3
Expected number of faults detected for different combinations of predictors in $\mathcal{M}_2$. Each row reports the range of *faults* corresponding to 94% probability and the mean on the posterior.

developers	fixed predictors	94% CI		mean
low experience	$experience = 0$	1,	8	4.02
high experience	$experience = 1$	1,	11	5.67
exploratory testing	$approach = 0$	6,	11	8.27
test-case testing	$approach = 1$	1,	2	1.42
exploratory and low	$approach = 0$ $experience = 0$	6,	8	6.92
exploratory and high	$approach = 0$ $experience = 1$	8,	12	9.61

with *more* faults being detected. In contrast, parameter β_e is likely positive. Since it models the effects of developer *experience* (another binary variable with 0 corresponding to low experience) it means that more experienced developers tend to find more faults. However, β_e's distribution in Fig. 3 has a non-negligible overlap with zero. Hence, we have weaker confidence in the significance of experience than we have in the significance of the testing approach.

Table 2 summarizes the posterior of all population-level parameters of $\mathcal{M}_2$. Through them, we can analyze other features of our fitted model. For instance, β_p is clearly positive, which means that test-case based testing is associated with a higher probability of detecting no faults.

Towards practical significance. By analyzing a posterior probability distribution we can move from statistical significance to practical significance. Since a full distribution is available, we are not limited to measuring probability intervals of the model's parameters, but we can also calculate probabilities of *outcomes*—that is what is the expected number of *detected faults* in different scenarios.

Concretely, we derive different marginal distributions from the posterior according to specific usage scenarios that we may want to analyze. For example, to estimate the expected number of *faults* that would be detected by a developer using exploratory testing ($approach = 0$) or by one using test-case based testing ($approach = 1$). Figure 4 (left figure) shows the ranges spanning a 94% probability interval, which not only confirm that exploratory testing is expected to be more effective but quantify the expected difference (roughly three times as many faults found).

A similar analysis comparing developers with different experience, in Fig. 4 (right figure), suggests instead that the fault-detection performance of developers with different experience can still be very similar as the two intervals have a large overlapping—even though more experienced developers are slightly more effective on average.

Table 3 reports the same information numerically, and extends the analysis to other scenarios such as comparing the effects of developer experience on the number of faults detected using exploratory testing. In Sect. 4 we will show these probabilities buttress a rigorous analysis of practical significance.

3.4 Bayesian vs. Frequentist Analysis of Significance

Overall, the high-level results of our Bayesian reanalysis are consistent with those of the original study [16]: exploratory testing performs significantly better than test-case based testing when looking at the number of faults found; the impact of experience is 'clearly' significant in the original study, while in the reanalysis one can question that. Unlike our reanalysis, the original study used frequentist statistics—similarly to previous analyses of the same processes [31], [32].

Even though the big picture does not change—and there is no reason it should—the distinctive features of Bayesian statistics make our reanalysis results more directly useful to assess *practical* significance in a robust and insightful way. The Bayesian emphasis on modeling entails that we could consider and compare different competing statistical models on the grounds of their characteristics and perfor-

mance. More important, features of the chosen model $\mathcal{M}_2$ strengthen our understanding of the studied phenomenon: zero-inflation indicates that a significant portion of the trials found no faults; a varying-effects term accounts for the significant individual differences among developers. These model features do not affect the conclusion that one testing approach performs significantly better than the other; but they help quantify the relative weight of different factors more precisely and in terms of phenomena in the actual problem domain.

The other key feature of Bayesian analysis that strengthens our understanding of the studied phenomena is the capability of deriving probability distributions of the outcome variables. In our case, we derived the expected number of *faults* a pool of developers with certain characteristics would find. This expresses the "significance" of a certain difference between treatments in terms of measures that are relevant in practice in the domain of testing processes.

As we show in the next section, this feature is also the basis to combine a statistical model of the data with features of how humans assess probabilities and make decisions—leading to an all-round assessment of practical significance.

4 Arguing Practical Significance

We want to frame practical significance in a way that it supports *choices* between alternatives. Based on our case study, we imagine a manager who can select a testing approach (exploratory or test-case based) and developers with lower or higher experience. To support the manager's choices, we quantify the expected *utility* of each choice: a monetary value that approximates the gains (if positive) or losses (if negatives) that are likely to derive from that choice.

Even though our example can be seen as simplistic, it is grounded on a need that partners in industry had, when deciding between two test techniques [16]. However, more complex decisions can also be made, in particular when involving multiple stakeholders with conflicting views. But, in those cases one would need to look at game theory as the underlying decision-making framework [33]. In our case, one could rather see an alternative, and if you will, more straightforward, approach.

4.1 Value and Weight Functions

A simple approach would just use the probabilities of different outcomes (computed from $\mathcal{M}_2$) to average the value $\nu(x)$ of each possible outcome x of decision X. Instead, we use cumulative prospect theory (CPT) to "adjust" the probabilities so that they reflect how humans are likely to perceive alternatives. The expected "subjective" utility $\mathbb{E}_U(X)$ of decision X is given by the weighted average,[10]

$$\mathbb{E}_U(X) \;=\; \sum_{x \in X} w(P(x)) \cdot \nu(x). \tag{13}$$

where w is a *weight function* that adjusts the probability $P(x)$ of each outcome.

In Eq. (13) above, P can be computed from our posterior probability distribution. As weight function we can use a standard function proposed by Tversky and Kahneman [14]. The idea of this function is that intermediate probabilities are flattened as if they were similar, whereas probabilities closer to the extremes are accentuated. This reflects the human perception that tends to conflate small probabilities with impossibility, and large probabilities with certainty.

We are left with providing a definition of $\nu(x)$ for each possible outcome x. In our case study, outcome x corresponds to *faults* $= x$, that is x faults detected during a testing session. We can associate a monetary value to each outcome based on the costs and benefits that come with it:

$$\nu(\textit{faults} = x) \;=\; (S \cdot x) - (C \cdot h) \tag{14}$$

where S are the savings for each fault found, C is the hourly pay of a developer, and h is the number of hours of a testing session. Since our study's experiment involved 2×90 minute sessions, we set $h = 3$. The manager of a company could suggest values for S and C. In this section, we take $S = \$150$, $C^- = \$100$ for low-experience developers, and $C^+ = \$200$ for high-experience developers. These values are realistic values for a small software company in Sweden.

4.2 Utility of Different Choices

Consider three possible choices a manager has to make, and compute the expected utility according to Eq. (13).

approach: the manager chooses whether developers use exploratory testing or test-case based testing; the available developers are a mix of low-experience and high-experience

experience: the manager chooses whether to hire low-experience or high-experience developers to do testing; they will use a mix of exploratory and test-case based testing

exploratory: the manager chooses whether to hire low-experience or high-experience developers to do exploratory testing

Whenever there is a mix of options, we assume it reflects averages in our data (i.e., the sample is representative of the population).

4.2.1 Choosing the Approach

In this scenario, **approach** is the choice is between using *exploratory* testing or *test-case* based testing. We use $\overline{C} = \$134.38$ as hourly developer cost. This is the average per-person cost in a pool of 35 developers—23 with low experience and 12 with high experience as in the dataset.

According to Eq. (13), the expected utility of choosing exploratory testing is $\mathbb{E}_U(\textit{exploratory}) = \$454.3$; the expected utility of choosing test-case based testing is $\mathbb{E}_U(\textit{test-case}) = \$8.0$. Hence, it is clear exploratory testing is likely to bring a much higher value.

In addition to expected utility, we can compute the utility value associated with outcomes with a certain probability. As is customary in CPT, we split the probability unit interval into three parts $3\%, 94\%, 3\%$ and compute utility for each sub-interval.[11] Figure 5 shows the results for the scenario's

10. If the outcomes are a continuum, the average should be computed using an integral and cumulative probabilities.

11. We like to present the tails of a distribution, and the tails should be much smaller than the bulk of the distribution.

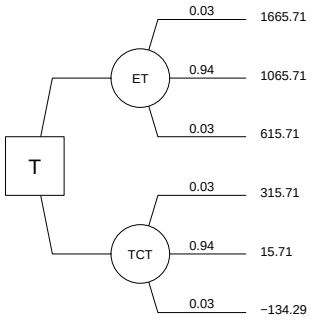

Fig. 5. Utility for different choices in scenario **approach**: the manager chooses whether developers use exploratory testing (ET) or test-case based testing (TCT).

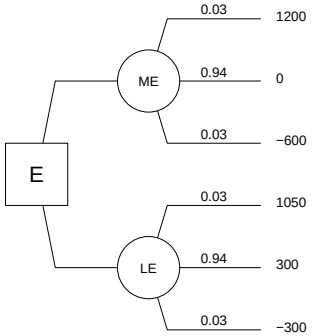

Fig. 6. Utility for different choices in the scenario for **experience**: the manager chooses whether to hire developers with low (LE) or high (ME) experience.

approach: using test-case based testing would lead to significant gains ($315.71) with 3% probability; to significant losses ($-\$134.29$) also with 3% probability; and to modest gains ($15.71) in the vast majority of cases (94% probability). In contrast, using exploratory test-case based testing makes any losses vanishingly unlikely to happen.

4.2.2 Choosing the Experience

In this scenario, **experience** is the choice between using developers with *low* or *high* experience. According to Eq. (13), the expected utility of hiring low-experience developers is $\mathbb{E}_U(low) = \$304.50$, and the expected utility of hiring high-experience developers is $\mathbb{E}_U(high) = \$18$. This means that, with the combination of exploratory and test-case based testing seen in the case study, high-experience developers are not worth what they cost; low-experience developers achieve a more favorable trade-off. Figure 6 shows the breakdown of utility for different ranges of probability in the scenario regarding **experience**.

Remember that these results depend on the assumptions about the cost of undetected faults and the hourly cost of developers. If, for example, each detected fault would bring a gain of $\$1000$—rather than $\$150$ as we have assumed so far—the expected utility of hiring low-experience developers would become $6700, smaller than the expected utility $9400 of hiring high-experience developers. In other words, if finding as many faults as possible is critical, even the modest performance advantage of high-experience developers may be worth the higher costs.

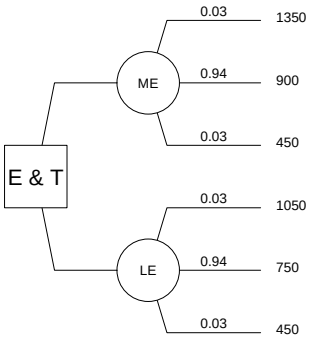

Fig. 7. Utility for different choices in the scenario **exploratory**: the manager chooses whether to hire developers with low (LE) or high experience (ME) to perform exploratory testing.

4.2.3 Choosing the Experience, Given Exploratory Testing

In this scenario, **exploratory** is the choice, again, between using developers with *low* or *high* experience, but all of them use exploratory testing. In this case, the expected utility of hiring low-experience developers is $\mathbb{E}_U(low + exploratory) = \$750$; the expected utility of hiring high-experience developers is $\mathbb{E}_U(high + exploratory) = \$900$. More experienced developers are worth their higher pay in this scenario; but the difference is small, and hence a manager may also include low-experience developers if high-experience ones turn out to be hard to find or unavailable. Figure 7 shows the breakdown of utility for different ranges of probability in the scenario of **experience**.

4.3 Usage of CPT to Foster Practical Significance

In summary, we used CPT to evaluate the practical impact of our investigation with respect to both the testing approach and the experience of testers. We designed three choice problems related to the factors investigated in our experiment, and provided different outcomes for each choice problem, along with a suggested decision based on the utility of those choices.

Cumulative prospect theory allows us to explicitly discuss and present practical significance by creating choice problems based on Bayesian analysis. The original study did not include a discussion about the practical impact of its results in connection to the costs of a fault or other contextual data from industry. Using our approach in a hypothetical context with costs for faults and salaries, we could instead illustrate how statistical predictions and CPT support decision making.

We should be clear that our values for costs of faults and salaries are arbitrarily chosen—though based on estimates of the Swedish job market—and could change dramatically according to the software's domain, the characteristics of the company, and many other factors [34]. Our only assumption is that it is somewhat possible to estimate such costs by collecting the necessary information from research, practitioners, and common knowledge.[12] Whenever this assumption is satisfied, this approach is applicable.

12. When a precise estimate is unavailable, the model could also incorporate *uncertainty* in the estimates as probability intervals, and calculate how the uncertainty in the estimates translates to uncertainty in the outcomes. Interested readers can try this out in our analysis using the supplementary material.

5 Validation with Practitioners

Section 4 demonstrated our approach of applying cumulative prospect theory (CPT) on a posterior probability distribution to argue significance in terms of costs and benefits in practice. On the other hand, statistics alone can—and often is—used to discuss significance from a statistical perspective. To ascertain whether practitioners and domain experts do indeed find a presentation of significance in terms of utility clearer than a traditional statistical analysis, we conducted a qualitative empirical validation. The rest of this section describes its design and results.

5.1 Validation Design

The overall goal of the validation is to compare two different ways of framing significance results: 1) using standard (frequentist) statistical techniques; and 2) using our combination of Bayesian analysis and CPT. The comparison is from the point of view of stakeholders using statistically significant results to make a decision.

Participants. We contacted 22 managers working at two large companies in Sweden who agreed to participate in this validation. The participants are a convenience sampling, and come from both middle (15 participants) and upper (7 participants) management positions. None of the participants were involved in this research, nor in the original analysis of testing approaches [16].

Instruments. We provided all participants with two summaries of the comparison between exploratory and test-case based testing: 1) a summary based on the statistics and results of the original study [16] (using frequentist statistics); 2) a summary based on the statistics and results of the present paper's re-analysis (using Bayesian statistics and CPT). Participants were asked to read the summaries and answer two yes/no questions about each of the summaries:

approach: Based on the presented information, would you use exploratory testing?

experience: Based on the presented information, would you use more experienced testers?

After answering each question with yes or no, participants also rated, on a 1–5 Likert scale, how confident they were in their answer (1: not confident at all; 5: completely confident), i.e., the construct used to assess which approach was better.

The questionnaire material was prepared by two of the authors and validated by two other authors.[13] The hourly costs for junior/senior developers and the costs for a bug was set according to the participating companies' input, i.e., \$60, \$70, and \$10,000, respectively. The questionnaire was then used at two workshops. One of the authors was available to answer any questions about the questions or the summaries.

We only retained 18 questionnaires out of 22—those that had all required information filled in.

5.2 Validation Results

The results of the questionnaires are summarized using diverging bar charts in Figs. 8–9. Each horizontal bar is centered on the middle point of the Likert scale (value 3) and spans the percentage of participants answering with low (values 1 and 2 on the Likert scale spanning the left of the middle point) or high (values 4 and 5 on the Likert scale spanning the right of the middle point).

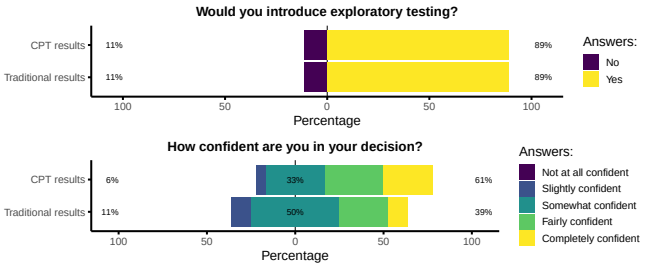

Fig. 8. Responses from 18 subjects about question **approach**: "Would you use exploratory testing?"

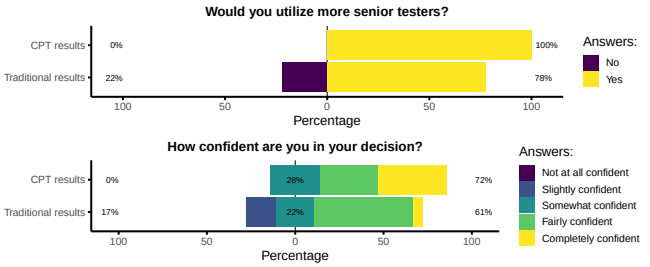

Fig. 9. Responses from 18 subjects about question **experience**: "Would you use more experienced testers?"

Figure 8 summarizes the results concerning **approach**. In this case, the decisions made by participants do not depend on how the data is presented. However, the summary based on our approach combining Bayesian models and CPT tends to increase the confidence in the decision: 61% of participants were fairly or completely confident—instead of 39% with the "traditional" summary.

Figure 9 summarizes the results concerning **experience**. In this case, the summary based on our approach convinced all participants to choose more experienced testers, whereas the "traditional" statistical summary convinced only 78% of them. As in answers to the other question, Bayesian models and CPT tend to increase the confidence in the decision: 39% of participants were completely confident—instead of 6% with the "traditional" summary. In both cases, confidence in answering this question was higher than in answering question **approach**.

By analyzing the recorded discussions after the session we received a qualitative, richer, view on the perception the subjects had concerning the two approaches, which we define as Case 1 (the original study's results) and Case 2 (the results we propose in this paper).

First, it is worthwhile to note that the subjects were only asked, in a group session, two questions to start the discussions concerning Case 1 and 2: How do you make sense of the results in [Case 1 or Case 2]? Second, the subjects spent approximately twice as much time discussing Case 2. Quite simply, the subjects exhausted Case 1 much sooner. If we consider two very representatives quotes concerning Case 1, we will see why this could be the case:

13. The questionnaire is available in the replication package.

> Quite difficult to understand. In particular the difference between more or less experienced developers was very unclear in [Case 1].

and,

> I was more looking into the summary [the conclusion] ... it said it was no significant difference [between more or less experienced developers].

The first quote, which is representative of many comments, indicates that Case 1 is more difficult to understand. The second quote could be an indication that Case 1's use of frequentist statistics, where we fall back on the arbitrary 95% significance level, did not encourage them to look further.

Concerning Case 2 the following quotes paint another picture:

> Very clear. It becomes so easy [...] makes me instead think about other things [...] how was data collected?

,

> The example [Case 2] made me start thinking more about other things that could affect the effectiveness [of the two techniques].

, and

> We can look at a research paper and interpret it from our [the company's] perspective.

This validation provides some preliminary evidence that presenting significance results using a combination of Bayesian statistics and cumulative prospect theory might help increase the confidence in a decision between alternatives—even when the outcome of a decision is not greatly affected since the underlying data remains the same.

6 Discussion

We have presented a method for how to assess and argue for the practical significance of empirical software engineering results. By combining Bayesian Data Analysis (BDA) with systematic evaluation of outcomes, using Cumulative Prospect Theory (CPT), different scenarios can be simulated and compared. While the posterior probability distribution from BDA summarizes the (scientific) knowledge gained by the research, the scenario simulation can help practitioners connect to concrete situations and, thus, increases the practical significance of the research while also informing decisions. This is supported by our evaluation with managers at two companies in the software industry.

Applying this approach in other settings is straightforward. It applies anywhere there is a quantifiable outcome for a technique, e.g., effort estimation or bug detection. The caveat is that measurements are only as good as their construct validity and the precision of the measurement. In situations where it is very hard to quantify the value of different outcomes it will be hard to apply the technique.

To apply the approach, we follow the steps outlined in Fig. 1. To choose value and weighting functions, an initial approach is to choose a function that is a simple linear/exponential/sigmoid, as these three categories capture most of the valuation functions we have seen. Asking practitioners for three or four values and selecting between these three categories should then allow a suitable value function to be fitted. Other approaches are beyond the scope of this article but could include fuzzy logic, for example [35].

In situations where the data and/or model is more complex than the case we re-analysed here, e.g. when there are more factors or they interact in affecting the dependent variable(s), we argue that the proposed methodology should be relatively more valuable. This is because humans would have a relatively harder time judging the scenarios or understanding the effects if there are complex or non-linear interactions. Simulation and concrete metrics as a basis for comparison thus will be more important.

There is a risk that the simplicity encouraged by the use of CPT or, really, any method that considers few factors and assigns them simple numerical values, would lead software engineers and managers to not consider the many factors that are critical to real-world decision making [36]. For an obvious example, in the case studied here, it would not be wise for a manager to simply prefer a more experienced engineer over a less experienced, without considering how they would fit into the development team. However, we argue that practitioners understand this and do not expect research to fully cater to their everyday situation. Also, we argue that this is a risk with the underlying study itself regardless of the way the collected data is then analyzed. However, we still advise caution in the claims that one makes based on the type of analyses proposed here; the additional clarity offered by summarizing specific outcomes in simple numerical values should not be misused.

In closing, we note that there is research on how to present probabilities and research results for better effect [37] and that there has also been several criticisms of CPT [36], [38]. We leave it to future work to explore alternatives to CPT, in this regard.

7 Threats to Validity

The canonical structure used to discuss threats to validity mainly targets experimental design and sub-sequence statistical analysis; hence it does not fit this work very well. Instead, we present our analysis of threats to validity as "bad smells" of analysis work [39].

1) **Not interesting.** (Research that has negligible software engineering impact.) The problem of analyzing practical significance is itself relevant and significant in practice, as demonstrated by previous work on statistical analysis that we summarize in Sect. 2.
2) **Not using related work.** (Unawareness of related work concerning RQs and SOA.) Section 2 also discusses the previous approaches to arguing practical significance, and their limitations.
3) **Using deprecated and suspect data.** (Using data out of convenience.) Our case study is a reanalysis of existing data, which was recently analyzed in a peer-reviewed publication [16].
4) **Inadequate reporting.** (Partial reporting, e.g., only means.) One of the key features of Bayesian statistics, which we propose as a basis for a more thorough analysis of practical significance, is its focus on modeling complete probability distributions instead of only point estimates.

5) **Under-powered experiments.** (Small effect sizes and little theory.) While we did not perform power analysis explicitly, model comparison and other diagnostic techniques of Bayesian data analysis are useful to choose models based on their effectiveness in practice and in theory.
6) **$p < 0.05$ and all that.** (Abuse of null hypothesis testing.) Null-hypothesis testing is manifestly in contrast to the statistical modeling approach we propose.
7) **Assumptions of normality and equal variances.** (Impact of outliers and heteroscedasticity.) In our case study, we use a (multi-level) generalized linear model with a Poisson likelihood. Additionally, we employed multi-level modelling, which helps avoid overfitting and handle outliers appropriately.
8) **Not exploring stability.** We did not discuss them in the paper for brevity, but we ran all recommended diagnostics of Bayesian analysis to ensure that there are no stability problems in the fitted models used in our study, this includes prior predictive checks.
9) **No data visualization.** We used data visualization to complement numeric data—focusing on the key measures of interest.
10) **Not tuning, not exploring simplicity, and not justifying choice of learner (overfitting).** In our case study, we explored two competing models, and chose one based on both theoretical considerations and the information criterion concerning out-of-sample predictive performance. Thus, when tuning our models, we considered models of different complexity, and we employed techniques to reduce the chance of overfitting.

8 Conclusions

We presented an approach to argue practical significance in empirical software engineering. Our approach develops a Bayesian model of the data, then it applies cumulative prospect theory on top of the model to incorporate a quantitative notion of utility and how probabilities are subjectively perceived by humans facing a decision. We demonstrated the approach on data from a previously published study comparing exploratory testing to test-case based testing [16].

To ascertain whether our presentation of statistically significant results is indeed accessible to practitioners, we conducted a small-scale empirical validation where we asked managers to report their confidence in decisions concerning the study's results. More precisely, we compared decisions informed by our presentation of practical significance to decisions based on the original frequentist statistical analysis [16]. Our combination of Bayesian statistics and cumulative prospect theory tends to increase the decision makers' confidence.

In future work, we plan to demonstrate our approach on larger case studies, and to perform more extensive validations of its usefulness in practice.

Replication Package

A replication package—including all data analyzed in the paper, scripts used to perform the analysis, and a Docker image with the tools to run the scripts—is available at:

https://github.com/torkar/docker-b3

Acknowledgments

We express our thanks to Paul-Christian Bürkner, Aki Vehtari, and Jonah Gabry, for the discussion on priors, and pointing us to literature discussing the choice of priors [40].

References

[1] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2008.
[2] K.-J. Stol, P. Ralph, and B. Fitzgerald, "Grounded theory in software engineering research: A critical review and guidelines," in *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, ser. ICSE '16, May 2016, pp. 120–131.
[3] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, and B. Regnell, *Experimentation in software engineering*. Springer, 2012.
[4] A. Arcuri and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering," in *Proceedings of the 33rd International Conference on Software Engineering*, ser. ICSE '11. New York, NY, USA: ACM, 2011, pp. 1–10.
[5] C. A. Furia, R. Feldt, and R. Torkar, "Bayesian data analysis in empirical software engineering research," *Transactions on Software Engineering*, pp. 1–1, 2019, accepted for publication.
[6] F. G. de Oliveira Neto, R. Torkar, R. Feldt, L. Gren, C. A. Furia, and Z. Huang, "Evolution of statistical analysis in empirical software engineering research: Current state and steps forward," *Journal of Systems and Software*, vol. 156, pp. 246–267, 2019.
[7] C. Tantithamthavorn and A. E. Hassan, "An experience report on defect modelling in practice: Pitfalls and challenges," in *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, ser. ICSE-SEIP '18. New York, NY, USA: ACM, 2018, pp. 286–295.
[8] M. Storey, E. Engstrom, M. Höst, P. Runeson, and E. Bjarnason, "Using a visual abstract as a lens for communicating and promoting design science research in software engineering," in *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, Nov 2017, pp. 181–186.
[9] K. Petersen and E. Engström, "Finding relevant research solutions for practical problems: The SERP taxonomy architecture," in *Proceedings of the 2014 International Workshop on Long-term Industrial Collaboration on Software Engineering*, ser. WISE '14. New York, NY, USA: ACM, 2014, pp. 13–20.
[10] J. Siegmund, N. Siegmund, and S. Apel, "Views on internal and external validity in empirical software engineering," in *Proceedings of the 37th International Conference on Software Engineering - Volume 1*, ser. ICSE '15. Piscataway, NJ, USA: IEEE Press, 2015, pp. 9–19.
[11] E. Engström, K. Petersen, N. B. Ali, and E. Bjarnason, "SERP-test: A taxonomy for supporting industry–academia communication," *Software Quality Journal*, vol. 25, no. 4, pp. 1269–1305, 2017.
[12] R. McElreath, *Statistical rethinking: A Bayesian course with examples in R and Stan*. CRC Press, 2015.
[13] A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin, *Bayesian data analysis*, 3rd ed., ser. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 2013.
[14] A. Tversky and D. Kahneman, "Advances in prospect theory: Cumulative representation of uncertainty," *Journal of Risk and Uncertainty*, vol. 5, no. 4, pp. 297–323, Oct 1992.
[15] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk," *Econometrica*, vol. 47, no. 2, pp. 263–291, 1979.
[16] W. Afzal, A. N. Ghazi, J. Itkonen, R. Torkar, A. Andrews, and K. Bhatti, "An experiment on the effectiveness and efficiency of exploratory testing," *Empirical Software Engineering*, vol. 20, no. 3, pp. 844–878, Jun 2015.
[17] A. M. Bayoumi and D. A. Redelmeier, "Decision analysis with cumulative prospect theory," *Medical Decision Making*, vol. 20, no. 4, pp. 404–411, 2000.
[18] C. Heath and A. Tversky, "Preference and belief: Ambiguity and competence in choice under uncertainty," *Journal of Risk and Uncertainty*, vol. 4, no. 1, pp. 5–28, Jan 1991.
[19] M. Shaw, "Prospects for an engineering discipline of software," *Software*, vol. 7, no. 6, pp. 15–24, 1990.
[20] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *Transactions on Software Engineering*, vol. 28, no. 8, pp. 721–734, 2002.

[21] K.-J. Stol and B. Fitzgerald, "The ABC of software engineering research," *Transactions on Software Engineering and Methodology*, vol. 27, no. 3, pp. 1–51, Sep. 2018.
[22] P. J. Runkel and J. E. McGrath, *Research on human behavior*. Holt, Rinehart, and Winston, Inc., 1972.
[23] C. Williams, M.-A. Storey, N. A. Ernst, A. Zagalsky, and E. Kalliamvakou, "Methodology matters: How we study socio-technical aspects in software engineering," *arXiv preprint arXiv:1905.12841*, 2019.
[24] R. K. Yin, *Case study research: Design and methods*, 3rd ed. Sage Publishing, 2002.
[25] R. Wieringa, *Design science methodology: For information systems and software engineering*. Springer, 2014.
[26] J. Correll, C. Mellinger, G. H. McClelland, and C. M. Judd, "Avoid Cohen's 'small', 'medium', and 'large' for power analysis," *Trends in Cognitive Sciences*, vol. 24, no. 3, pp. 200–207, Mar. 2020.
[27] V. B. Kampenes, T. Dybå, J. E. Hannay, and D. I. Sjøberg, "A systematic review of effect size in software engineering experiments," *Information and Software Technology*, vol. 49, no. 11-12, pp. 1073–1086, Nov. 2007.
[28] A. Gelman, "Prior distributions for variance parameters in hierarchical models," *Bayesian Analysis*, pp. 515–533, 2006.
[29] B. Carpenter, A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell, "Stan: A probabilistic programming language," *Journal of Statistical Software*, vol. 76, no. 1, pp. 1–32, 2017.
[30] A. Vehtari, A. Gelman, and J. Gabry, "Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC," *Statistics and Computing*, vol. 27, pp. 1413–1432, 2017.
[31] J. Itkonen and M. V. Mäntylä, "Are test cases needed? Replicated comparison between exploratory and test-case-based software testing," *Empirical Software Engineering*, vol. 19, no. 2, pp. 303–342, Apr 2014.
[32] J. Itkonen, M. V. Mäntylä, and C. Lassenius, "The role of the tester's knowledge in exploratory software resting," *Transactions on Software Engineering*, vol. 39, no. 5, pp. 707–724, May 2013.
[33] J. Nash, "Non-cooperative games," *Annals of Mathematics*, vol. 54, no. 2, pp. 286–295, 1951.
[34] B. Boehm and V. R. Basili, "Software defect reduction top 10 list," *Computer*, vol. 34, no. 1, pp. 135–137, Jan. 2001.
[35] G. Büyüközkan and O. Feziοğlu, "A fuzzy-logic-based decision-making approach for new product development," *International Journal of Production Economics*, vol. 90, no. 1, pp. 27–45, Jul. 2004.
[36] M. Nwogugu, "A further critique of cumulative prospect theory and related approaches," *Applied mathematics and computation*, vol. 179, no. 2, pp. 451–465, 2006.
[37] D. N. Wijeysundera, P. C. Austin, J. E. Hux, W. S. Beattie, and A. Laupacis, "Bayesian statistical inference enhances the interpretation of contemporary randomized controlled trials," *Journal of clinical epidemiology*, vol. 62, no. 1, pp. 13–21, 2009.
[38] G. Gigerenzer, "How to make cognitive illusions disappear: Beyond 'heuristics and biases'," *European review of social psychology*, vol. 2, no. 1, pp. 83–115, 1991.
[39] T. Menzies and M. Shepperd, "'bad smells' in software analytics papers," *Information and Software Technology*, vol. 112, pp. 35–47, 2019.
[40] D. Simpson, H. Rue, A. Riebler, T. G. Martins, and S. H. Sørbye, "Penalising model component complexity: A principled, practical approach to constructing priors," *Statistical Science*, vol. 32, no. 1, pp. 1–28, 02 2017.